Supporting conceptual knowledge capture through automatic modelling: A preliminary progress report

Liem, J.; Buisman, H.; Bredeweg, B.

Publication date
2008

Published in
22nd International Workshop on Qualitative Reasoning

Supporting Conceptual Knowledge Capture Through Automatic Modelling: A Preliminary Progress Report

Jochem Liem and **Hylke Buisman** and **Bert Bredeweg**
Human Computer Studies Laboratory, Informatics Institute, Faculty of Science,
University of Amsterdam, The Netherlands. Email: {jliem,bredeweg}@science.uva.nl, hbuisman@gmail.com

Abstract

Building qualitative models is still a difficult and lengthy endeavour for domain experts. This paper discusses progress towards an automated modelling algorithm that learns Garp3 models based on a full qualitative description of the system's behaviour. In contrast with other approaches (Bridewell et al. 2008; Bratko and Šuc 2004), our algorithm attempts to learn the causality that explains the system's behaviour. The algorithm achieves good results when recreating four well-established models.

Introduction

In this paper we focus on the ground work required to advance towards an automated modelling program. The input is considered to have a qualitative representation, i.e. a state graph that represents the possible situations that can emerge from a system, and the values of the quantities in each situation. Furthermore, the input is assumed to have no noise nor any inconsistencies. The completed algorithm is envisioned to support researchers in articulating their conceptual understanding. As such it will help to establish theories that explain the phenomena provided as input data.

QR Model and Simulation Workbench: Garp3

The automatic model building algorithm is implemented in Garp3[1] (Bredeweg et al. 2006). Garp3 allows modellers to represent their knowledge about the structure and the important processes in their system as *model fragments*, which can be considered formalisations of the knowledge that applies in certain general situations.

Next to model fragments, different *scenarios* can be modelled. These represent specific start states of a system. Garp3 can run simulations of models based on a particular scenario. The result of such a simulation is a state graph, in which each state represents a particular possible situation of the system, and the transitions represent the possible ways a situation can change into another.

The simulation engine takes a scenario as input, and finds all the model fragments that apply to that scenario. The consequences of the matching model fragments are added to the scenario to create a state description from which new knowledge can be inferred such as the derivatives of quantities. Given the completed state description, the possible successor states are inferred. The complete state graph is generated by applying the reasoning to the new states.

In Garp3 the structure of a system is represented using *entities* (objects) and *configurations* (relations). For example, a lion hunting on a zebra would be represented as two entities (lion and zebra) and a configuration (hunts).

Quantities represent the features of entities and agents that change during simulation. A quantity has a magnitude and a derivative, which represent its current value and trend. The magnitude and derivative are each defined by a quantity space that represents the possible values the magnitude and the derivative can have. Such a quantity space is defined by a set of alternating *point* and *interval* values.

We use $M_v(Q_1)$ to refer to the current value of the magnitude of a quantity. $M_s(Q_1)$, the sign of the magnitude, indicates whether the magnitude is positive, zero or negative ($M_s(Q_1) \in \{+, 0, -\}$). $D_v(Q_1)$ refers to the current value of the derivative of a quantity, which has a value from the predefined derivative quantity space ($D_v(Q_1) \in \{-, 0, +\}$). $D_s(Q_1)$ refers to the current sign of a derivative. Note that the predefined values of derivatives completely correspond to the possible signs of the derivative.

Causality

Garp3 explicitly represents causality using indirect and direct influences. Direct influences are represented as $Q_1 \overset{I_+}{\rightarrow} Q_2$. Influences can be either positive (as above) or negative. The positive influence will increase $D_v(Q_2)$ if $M_s(Q1) = +$, decrease it if $M_s(Q1) = -$, and have no effect when $M_s(Q1) = 0$. For a negative influence, it is vice versa.

The indirect influences, called *proportionalities*, are represented as $Q_1 \overset{P_+}{\rightarrow} Q2$. Similar to influences, proportionalities can be either positive or negative. The positive proportionality will increase $D_v(Q_2)$ if $D_s(Q_1) = +$, have no effect if it is stable, and decrease if it is below zero. For a negative proportionality, it is vice versa.

Other Behavioural Ingredients

Other behavioural ingredients in Garp3 are operators, inequalities, value assignments and correspondences. Opera-

[1] http://www.garp3.org

tors (+ and -) are used to calculate the magnitude value of quantities (e.g. $Q_1 - Q_2 = Q_3$, to indicate $M_v(Q_1) - M_v(Q_2) = M_v(Q_3)$). Inequalities can be placed between different model ingredient types: (1) magnitudes ($M_v(Q_1) = M_v(Q_2)$), (2) derivatives ($D_v(Q_1) < D_v(Q_2)$, (3) values $Q_1(point(Max)) = Q_2(point(Max))$, (4) operator relations ($M_v(Q_1) - M_v(Q_2) < M_v(Q_3) - M_v(Q_4)$, (5) combinations of the 1, 2, 3 and 4 (although only between either magnitude or derivative items). Value assignments simply indicate that a quantity has a certain qualitative value ($M_v(Q_1) = Q_1(Plus)$). Finally, correspondences indicate that from certain values of one quantity, values of another quantity can be inferred. There are quantity correspondences ($Q_1 \overset{Q_{qs}}{\leftrightarrow} Q_2$) and value correspondences ($Q_1(Plus) \overset{Q_v}{\rightarrow} Q_2(Plus)$), which can both be either directed or undirected. The value correspondence indicates that if $M_v(Q_1) = Q_1(Plus)$, $M_v(Q_2) = Q_2(Plus)$. If the value correspondence is bidirectional, the reverse inference is also possible. Quantity correspondences can be considered a set of value correspondences between each consecutive pair of the values of both quantities. There are also inverse quantity space correspondences ($Q_1 \overset{Q_{qs}^{-1}}{\leftrightarrow} Q_2$) that indicate that the first value in Q_1 corresponds to the last value in Q_2, the second to the one before last, etc.

Algorithm Requirements and Approach

Assumptions and Scoping

The goal of the automatic model building algorithm is to take a state graph and a scenario as input, and generate the model that provides an explanation for the behaviour. Our approach focusses on the generation of causal explanation. Several assumptions are made to scope the work. In further research these assumptions can be alleviated. Firstly, input is assumed to have no noise or inconsistencies. Secondly, the state graph is assumes to be a full envisionment of the system's behaviour.

The second assumption is that a model can be build using a single model fragment. From a causal explanation point of view, it is reasonable to assume that influences and proportionalities never disappear, but that their effects are only nullified when quantities become zero or stable.

Thirdly, the algorithm is focussed on causal explanation and less on structure. Therefore, the entity hierarchy is assumed known.

Input and Output

The algorithm takes a complete state graph as input, which includes (1) the quantity names, (2) the quantity spaces, (3) the magnitudes and derivatives of the quantities in different states, (4) the observable inequalities, and (5) the state transitions. Furthermore, the algorithm is provided with the scenario that should produce the state graph, which consists of: (1) the entities, agents and assumptions involved, (2) structural information about the configurations between them, (3) the quantities and their initial values, and (4) the inequalities that hold in the initial state.

The output of the algorithm is one or more Garp3 qualitative models that explain (are consistent with) the input that can be immediately simulated.

Algorithm Design Approach

Since the semantics of model ingredients are formally defined, one would assume that it is clear how each ingredient manifests itself in the simulation results of a model. Otherwise, how would the implementation of a simulation engine have been possible? However, in practice, it is hard even for expert modellers to pinpoint the model ingredients that are responsible for certain (lack of) behaviour. This has several reasons. Firstly, a large set of inequalities are derived during qualitative simulation, of which the implications (other inequalities) are difficult to foresee. Secondly, the engine has a lot of intricacies (such as second order derivatives) which makes simulation results hard to predict. Thirdly, the branching in the state graph that results from ambiguity is difficult for people to completely envision.

For these reasons, an iterative algorithm design approach is chosen. Well-established models are ordered by complexity, and attempts are made to generate them using their own output. Each of the models requires a different (and increasingly large) set of considerations that must be dealt with.

The models chosen are Tree and Shade, Communicating Vessels, Deforestation, Population Dynamics and a set of other even more complex models[2]. Tree and Shade is the least complex model, containing only a few quantities, and causal dependencies, and no conditions, causal interactions, inequalities or operator relations. Communicating vessels is more complex, as it contains causal interactions, an operator, and inequalities. The deforestation model is different from the previous models as it contains many clusters linked to each other by proportionalities. Population dynamics is again more complex, due to the large amount of quantities, interactions and conditions.

Causality and Clusters

Causal Paths Important for the algorithm is the concept of *causal paths*. These are series of quantities connected by influences and proportionalities. A causal path is defined as a set of quantities that starts with an influence, and is followed by a arbitrary number of proportionalities. For example: $Q_1 \overset{I_+}{\rightarrow} Q_2 \overset{P_+}{\rightarrow} \ldots \overset{P_-}{\rightarrow} Q_{n-1} \overset{P_+}{\rightarrow} Q_n$. A quantity that has no proportionalities leading out of it ends the causal path. If a quantity has more than one proportionality leading out of it, multiple causal paths can be defined.

Since each influence represents the causal effect of a process, a causal path can be seen as the cascade of effects of a process. Given this perspective, certain successions of causal relations become unlikely. For example the causal path $Q_1 \overset{I_+}{\rightarrow} Q_2 \overset{I_+}{\rightarrow} Q_3 \overset{P_-}{\rightarrow} Q_4 \overset{I_+}{\rightarrow} Q_5$ would imply there are many active processes with short or no cascading effects.

Direction of Causality An important issue in scientific enquiry is the problem of correlation and causality. This

[2]The models are available at http://www.garp3.org

issue appears when trying to derive causal relations from the state graph. For example, $D_s(Q_1) = D_s(Q_2)$ can be an caused by $Q_1 \overset{P_+}{\rightarrow} Q_2$, $Q_2 \overset{P_+}{\rightarrow} Q1$, or even $Q_3 \overset{P_+}{\rightarrow} Q_1$ and $Q_3 \overset{P_+}{\rightarrow} Q_2$. Another example of this is in the communicating vessels model. Ideally, a model capturing the idea of a contained liquid would distinguish between Volume, Height and Bottom pressure, and have a particular causal account ($Volume \overset{P_+}{\rightarrow} Height \overset{P_+}{\rightarrow} Bottom\_pressure$). However, from the model's behaviour this causality may not be derivable, e.g. when the width of the containers doesn't change. As a result, the unique role of the quantities involved can only be inferred when the required variation for that is apparent in the input state-graph. Therefore, it is considered the modeller's responsibility to provide simulation examples which will allow the algorithm the make these critical distinctions. However, it can be considered the responsibility of the tool to indicate to the modeller that the causality between certain sets of quantities cannot be derived, and that examples showing these differences should be provided.

Clusters The algorithm makes use of a specific subset of causal paths called *clusters*. We define clusters as groups of quantities that exhibit "equivalent" behaviour. More specifically, a set of quantities constitute a cluster if their values either correspond ($Q_1 \overset{Q_{qs}}{\leftrightarrow} Q_2$) or inversely correspond ($Q_1 \overset{Q_{qs}^{-1}}{\leftrightarrow} Q_2$) to each other. Additionally, the corresponding derivatives should be equal ($D_v(Q_1) = D_v(Q_2)$), while inversely corresponding derivatives should be each other's inverse ($D_v(Q_1) = -D_v(Q_2)$).

A further constraint is that the corresponding quantities (not inverse) in a cluster must be completely equivalent. Therefore, $M_v(Q_1) = M_v(Q_2)$ must always hold. If an inequality holds between two quantities, they are considered not to belong to the same cluster.

During implementation it became obvious that clusters are not meaningful when quantities within a cluster belong to different entities. The reason for this originates from the idea of 'no function in structure'. Clusters involving multiple entities would integrate causality across individual structural units, which is undesired. Therefore, clusters can only contain quantities that belong to the same entity.

Quantities cannot be a member of more than one cluster. If Q_1 and Q_2 are in a cluster, and Q_1 and Q_3 are in a cluster, then Q_1, Q_2 and Q_3 must be in the same cluster. After all, if Q_1 and Q_2 have equivalent behaviour, and Q_1 and Q_3 have equivalent behaviour, by transitivity Q_2 and Q_3 have to exhibit equivalent behaviour.

Minimal Covering

The key requirement of the model building algorithm is that it explains the input behaviour. However, a second requirement is that the algorithm does not contain redundant dependencies. That is, the algorithm should return the minimal set of dependencies that explains the behaviour.

Two dependencies are considered *substitutionary* if they have the same effect on the simulation result (i.e. removing one of them would have no effect, however removing both would). *Complementary* dependencies are responsible for different aspects of the behaviour, and both have to be present to explain the data. The aim is to create an algorithm that is minimally covering, i.e. it should only contain complementary dependencies.

Algorithm

Finding Naive Dependencies

The goal of this step is to find (non-interacting) dependencies that are valid throughout the entire model (i.e. are not conditional). These causal relations are called *naive dependencies*, and provide the basis for the rest of the algorithm.

Consistency Rules Naive dependencies are identified using consistency rules. Each pair of quantities is checked using these rules to determine which of them potentially holds throughout the state graph. These rules make use of $M_v(Q_x), M_s(Q_x), D_v(Q_x), D_s(Q_x)$ of each quantity in a pair, and inequalities that hold between them. These statements are referred to as the *state information* of a quantity.

The consistency rules are derived from the semantics of the causal dependencies (see Section on Garp3). Examples of rules (that should hold throughout the state graph) are:

$$Q_1 \overset{I_+}{\rightarrow} Q_2 \text{ if } M_s(Q_1) = D_s(Q_2) \tag{1}$$

$$Q_1 \overset{I_-}{\rightarrow} Q_2 \text{ if } M_s(Q_1) = -D_s(Q_2) \tag{2}$$

$$Q_1 \overset{P_+}{\rightarrow} Q_2 \text{ if } D_s(Q_1) = D_s(Q_2) \tag{3}$$

$$Q_1 \overset{P_-}{\rightarrow} Q_2 \text{ if } D_s(Q_1) = -D_s(Q_2) \tag{4}$$

$$Q_1(V_x) \overset{Q_v}{\leftrightarrow} Q_2(V_y) \text{ if } M_v(Q_1) = Q_1(V_x) \implies M_v(Q_2) = Q_2(V_y) \tag{5}$$

$$Q_1 \overset{Q_{qs}}{\leftrightarrow} Q_2 \text{ if } \forall V_n (Q_1(V_n) \overset{Q_v}{\leftrightarrow} Q_2(V_n)) \tag{6}$$

Redundancy The set of dependencies that are found contain a lot of redundancy, i.e. many dependencies are substitutionary. For example, in the communicating vessels model $height \overset{P_+}{\rightarrow} pressure$, can be substituted by $pressure \overset{P_+}{\rightarrow} height$. The remainder of the algorithm selects the correct substitutionary groups, and uses the selected naive dependencies to derive more complex dependencies.

Determining Clusters

This step tries to determine clusters within the set of naive dependencies. The algorithm searches for quantities belonging to the same entity that exhibit equivalent behaviour, and tries to expand these candidate clusters by adding other quantities. Quantities are only added if they exhibit behaviour equivalent to the quantities already contained in the candidate cluster. If no more quantities can be added to a candidate cluster, the algorithm searches for other candidate clusters. By only considering models composed of clusters, the space of possible models is significantly reduced.

The validity of the candidate clusters is checked by determining if there is overlap between the clusters. All clusters that overlap are removed. An alternative would be to only remove clusters until no more overlap is present. However,

in practice no situations were encountered where this was desirable. An example of a found cluster is volume, height and pressure in the communicating vessels model. Note that these clusters are still missing influences (their actuators), these are determined later in the algorithm.

Generating Causal Paths

This step returns the possible causal orderings within clusters based on the cluster and naive dependencies sets. For each cluster a valid causal ordering is returned. Through backtracking other possible orderings are generated.

The quantities in a cluster can be either connected in a linear fashion ($Q_1 \overset{P_+}{\rightarrow} Q_2 \overset{P_+}{\rightarrow} Q3$) or using branching ($Q_1 \overset{P_+}{\rightarrow} Q_2$ and $Q_1 \overset{P_+}{\rightarrow} Q3$). The algorithm prefers linear branching, as branching does not often occur in practice. Additionally, the reduction of possible models is a significant advantage.

Another constraint that reduces the number of possible models is requiring clusters that belong to entities of the same type to have the same causal ordering. For example, if for one container $Volume \overset{P_+}{\rightarrow} Height \overset{P_+}{\rightarrow} Pressure$, than for other containers the same causal ordering must hold.

Actuating Clusters

The goal of the actuating clusters step is to connect clusters by identifying cluster actuations. This step takes the set of clusters with established causal orderings and the naive dependencies as input.

Clusters can either be actuated by another cluster, or act as an actuator itself. Furthermore, clusters can be connected by propagating an actuation. In a model, each cluster should take part in at least one of these kind of relations such that all clusters are related in a way. Otherwise, the model would include two separate non-interacting subsystems.

When one cluster actuates another, there is an influence relation between the two. Actuations are the most important form of connecting clusters, since these connections are the cause of change in the system. They are also the easiest to detect, due to the specific way influences manifest themselves in the state information. For this reason, actuations by influences are identified first. Two types of actuations though influences are distinguished: (1) *equilibrium seeking mechanisms* (ESM) and (2) *external actuators*.

Equilibrium Seeking Mechanisms ESMs are better known as *flows*, and are common in qualitative models. Flows cause two unequal quantities to equalize. The flow in the communicating vessels model has a non-zero value when the pressures in the two containers are unequal. The flow changes the volume of the containers, and thus the pressures to equalize. An ESM holds under the following two conditions: (1) $Q_1 = Q_2 - Q_3$, where $Q_1 \in C_1, Q_2 \in C_2, Q_3 \in C_3$, where the C's are clusters, and (2) $Q_4 \overset{I_-}{\rightarrow} Q_5$ and $Q_4 \overset{I_+}{\rightarrow} Q_6$, where $Q_4 \in C_1, Q_5 \in C_2, Q_6 \in C_3$. Note that in many cases $Q_1 = Q_4$, such as in the communicating vessels model.

Finding Calculus Relations The algorithm reduces the search space of finding ESMs using four constraints. Firstly, all quantities involved in the operator should be in different clusters (C_1, C_2 and $C3$ are unequal). Secondly, the set of naive dependencies should at least contain one influence from Q_1 (to serve as an actuation). Thirdly, both Q_2 and Q_3 would be at the end of the causal paths within their cluster, as in most cases this is the most meaningful interpretation. Finally, Q_2 and Q_3 are required to be of the same type, as only things of the same type can be subtracted.

External Actuators External actuators are causes of change more at the edges of the system compared to ESMs. To identify external actuators, the algorithm considers the influences in the naive dependencies that are not part of an ESM. Again, the minimal covering principle is applied to keep the number of dependencies to a minimum. As a result a cluster will never have more than one incoming actuation.

An actuation is only considered between C_1 to C_2 if the set of naive dependencies contains influences between each possible pair of quantities, such that $\forall Q_x \in C_1, \forall Q_y \in C_2 (Q_x \overset{I_+}{\rightarrow} Q_y)$. This removes the influences in the set of naive dependencies that are consistent with the behaviour by chance.

Alternative actuations are returned through backtracking. In the future, actuations may be chosen based on the structure of the system, as causal relations are more likely to occur parallel to structurally related entities.

Feedback A common pattern in qualitative models is feedback, which is a proportionality originating from the end of a causal path to the quantity actuating the causal path. Feedbacks are simply added if the naive dependencies contain one. The algorithm always adds feedback at the end of causal paths, since this is what happens in the investigated models. However, it could be the case that feedbacks from halfway a causal chain are also possible.

Linking Clusters by Propagation

This step connects the clusters that have not yet been connected through proportionalities, based on the naive dependencies. As with clusters, the causal ordering of the clusters cannot be distinguished. Therefore all possibilities are generated. Furthermore, the same design choices as with finding causal paths within clusters have been made. Only linear orderings of clusters are allowed (i.e. no branching).

Setting Initial Magnitudes

An influence has no effect if the magnitude of the quantity from which it originates is unknown. Therefore this step assigns initial values to quantities. Note that this step first generates a set of candidate assignments. When a value can be derived in another way than through assignment, it is removed from the set of value assignment candidates.

There are six ways to assign initial magnitudes. Firstly, if a value assignment for the quantity is present in the scenario, it requires no initialisation. Secondly, if the magnitude can be derived through a correspondence, the value is known. Thirdly, the result of a minus operator can be derived if an

inequality between its arguments is known. Based on the possible magnitudes of the result this inequality can be derived. Either this inequality is present in the scenario, or multiple inequalities should be made assumable by adding them as conditions in multiple model fragments. Garp3 automatically assumes unprovable values and inequalities if they are conditions in model fragments. Note that generating the conditional inequalities is currently beyond the scope of the algorithm, as it involves adding model ingredients to multiple model fragments. Fourthly, it is possible that a certain magnitude holds everywhere throughout the state graph. In this case, a value assignment is added as a (conditionless) consequence. Fifthly, a value could hold under certain conditions. However, this would require a value assignments with a conditional inequalities in separate model fragments. Therefore, it is currently beyond the scope of the algorithm. Finally, multiple model fragments could be created in which the magnitudes are present as conditions. Garp3 will generate the different states that would result by assuming each of the values. As with the conditional value assignments, having value assignments as conditions in multiple model fragments is currently beyond the scope of the algorithm.

Dependency Interactions

This step identifies dependency interactions (influences or proportionalities) based on the input behaviour. Dependency interactions are detected in the same way as naive dependencies, i.e. using a set of consistency rules. Interactions are not found as naive dependencies, as the individual dependencies are not consistent with the *entire* state graph (as an interaction results in more behaviour than a single dependency).

The algorithm assumes that the interaction consists opposing dependencies, such as birth vs. death and immigration vs. emigration.

Results[3]

The *tree and shade model* is successfully modelled by the algorithm. It returns two models, representing both possible directions of causality between Size and Shade. The initial magnitude assignment correctly finds a conditionless value assignment on Growth rate. The simulation results of these models are equivalent to that of the original model.

The dependencies of the *communicating vessels model* are correctly found. The algorithm returns 6 models; one for each possible causal ordering of amount, height and pressure. The algorithm also correctly identifies the ESM-based actuations of the clusters, by properly finding the min operator. Furthermore, all necessary causal dependencies and correspondences are identified. Model fragments that allow the assumption of initial values are missing (due to the fact that the algorithm generates a single model fragments). Adding an inequality between the pressures of the containers in the scenario allows the model to simulate without problems.

The *deforestation model* (containing entities 'Woodcutters', 'Vegetation', 'Water', 'Land' and 'Humans') is successfully modelled, including setting initial magnitudes using conditions. The simulation is equivalent to that of the original model. The causal ordering does differ, as it does not capture the branching of the causal paths in the original model. The resulting model however, is not considered wrong by experts, and is arguably better than the original. Over 2000 models are returned when generating all possible results, due to the many possible causal orderings.

The *population dynamics model* generates the correct models for the open and closed population scenarios. However, the initial values are not set.

The algorithm does not yet give correct results for the *heating/boiling*, *R-Star* and *Ants' Garden* models. For the heating model this is due to inequalities that hold under specific conditions, which are not taken care of in the algorithm. The R-Star and Ants' Garden are large models that resulted from specific research projects. As such, these models are an order of magnitude more complex than the other models. It is therefore not surprising that the algorithm in its current form cannot cope with them.

[3]For the models and references go to http://www.garp3.org

Conclusions & Future Work

This paper presents preliminary work towards an algorithm that automatically determines a Garp3 qualitative model, using an enumeration of all possible system behaviour as input. The algorithm uses consistency rules to determine the causal dependencies that hold within the system. Using the concept of clusters the search space is significantly reduced. Accurate results are generated for a set of well-established models. The results seem to suggests that it is possible to derive causal explanations from the behaviour of a system, and that model building support through an automatic model building algorithm is viable.

There are several algorithm improvements planned. The first improvement is to have a generalised representation for the ambiguity within and between clusters. That is, have a single representation for the complete model space. For simulation purposes an arbitrary instantiation can be chosen, as each one has an equivalent result. Secondly, the algorithm has to be improved to be able to create multiple model fragments in order to deal with conditional model ingredients. Thirdly, means have to be developed to be able to compare generated state graphs with the desired state graph.

References

Bratko, I., and Šuc, D. 2004. Learning qualitative models. *AI Mag.* 24(4):107–119.

Bredeweg, B.; Bouwer, A.; Jellema, J.; Bertels, D.; Linnebank, F.; and Liem, J. 2006. Garp3 - a new workbench for qualitative reasoning and modelling. In Bailey-Kellogg, C., and Kuipers, B., eds., *20th International Workshop on Qualitative Reasoning (QR-06)*, 21–28.

Bredeweg, B.; Salles, P.; Bouwer, A.; Liem, J.; Nuttle, T.; Cioaca, E.; Nakova, E.; Noble, R.; Rios Caldas, A. L.; Yordan, U.; Varadinova, E.; and Zitek, A. 2008. Towards a structured approach to building qualitative reasoning models and simulations. *Ecological Informatics* 3(1):1–12.

Bridewell, W.; Langley, P.; Todorovski, L.; and Džeroski, S. 2008. Inductive process modeling. *Machine Learning* 71:132.