VU '19 Qualitative Reasoning Report

Tycho Grouwstra
UvA ID: 6195180
tychogrouwstra@gmail.com

Gaurav Kudva
UvA ID: 12205583
gauravkudva2@gmail.com

March 2019

In this report, we explain how a container system such as a bathtub or a sink can be modelled through qualitative reasoning. We highlight the importance of the various assumptions made and describe the algorithm implemented to simulate such a system. We provide an analysis of the causal model / state-graph and how they can be interpreted to infer the various behaviours of the system.

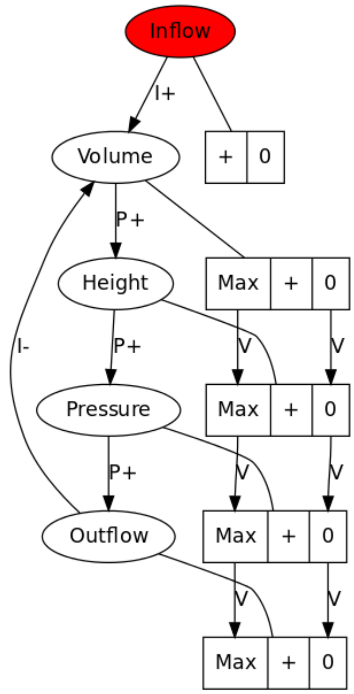
1 Causal Model

The full causal model and the expected state-graph are shown in figure 1. Here, the final two quantities *Pressure* and *Height* form the **full** model, whereas *Inflow*, *Volume* and *Outflow* comprise the **basic** model.

Table 1 summarizes the various correspondences and dependencies that exist in the system. Here, VC stands for value correspondence.

Note that out of these relationships, the relationships involving Pressure or Height are only part of the **full** model. The value correspondence rules [2] between Volume and Outflow, on the other hand, are only part of the **basic** model, as a shortcut to cut out the additional quantities.

The causal model shows a general overview of the various relationships and does not indicate a particular behaviour of the system. For this very reason, we have not included the derivatives for the involved quantities. The *Inflow* node has been marked with red to indicate its exogenous behaviour.



Causal Model (full version)

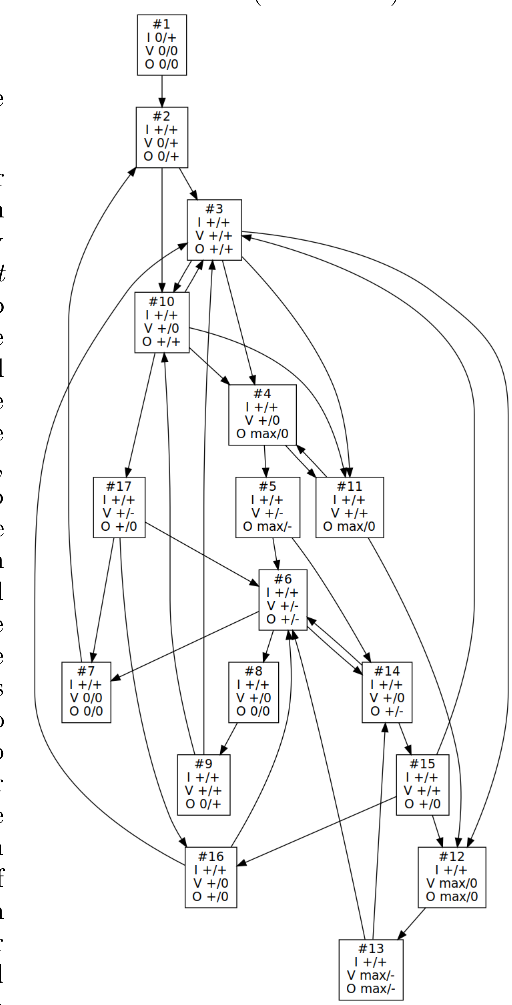


Figure 1: Full Causal Model and the Expected state graph for the basic model

2 Assumptions

Here, we present the assumptions chosen and how we think they might affect the generation of the state graphs.

We assume all quantity derivatives to take on values of either **positive**, **neutral**, or **negative**. We assume **zero** and **max** to be point-based values, but **plus** to be an interval-based value. Interval values may not change by their derivatives until any point values have already done so. In other words, while range magnitudes *might* change a step according to the derivative, point magnitudes *must* change a step according to the derivative. For the purpose of influence relationships [2], we assume **plus** and **max** to denote positive values, and **zero** a neutral value. These are used in influence relationships, as under a positive influence relation, a positive value will positively affect the derivative of another quantity, and vice versa for negative magnitudes and relationships denoting a negative correlation. More simply put, when water flows into our bathtub (positive inflow), the volume of our bathtub will increase. Among our derivative values, **plus** will be regarded as a positive derivative, **minus** as negative, and **zero** as neutral. Their use is two-fold: both for their effect on the magnitude values, as well as for the purpose of proportional relationships. Under the proportionality relationship, a derivative will influence the derivative of another quantity. These again encompass both positive and negative relationships. In our scenario, for example, changes in volume will cause changes in outflow. For our experiments we have assumed **value correspondences** to be **uni-directional**. We have assumed the value correspondences to tie the zero and max values in one direction, though not necessarily constraining the other value when the first one has a plus magnitude. This interpretation of the value correspondence rule will yield us a relatively higher number of total states. In terms of what influences what in our causal model, we follow a particular order of *Volume* → *Height* → *Pressure* → *Outflow*. Exogenous behavior may be modeled in various ways, including as steady, increasing, decreasing, as a parabola (positive or negative), sinusoidal, or random [2]. In order to simplify our scope, we have opted to use the **steady exogeny** model: we assume the Inflow in our model starts with a positive derivative. We assume derivatives are clipped when the magnitudes are at an extreme point. For example, when the magnitude of volume is at its maximum, then any derivative of plus will be truncated to the neutral value of zero (and vice versa for magnitudes at their minimum value of zero with a negative derivative of minus). We assume that magnitudes and derivatives abide by *continuity* [2]: they may not 'skip' intermediate steps. This simultaneously means that any magnitude quantity space is ordinal, i.e. zero comes first, then plus, then max. Our range of derivatives (minus, zero, plus) similarly exhibits an ordinal property. We have defined the order of calculation among the rules to be as follows:

1. Calculate potential next *magnitudes* based on the old magnitudes and derivatives (respecting that magnitudes in an *interval* may not change until any at a *point* have already changed, while also respecting *value correspondence*);
2. Calculate potential new derivatives based on the old derivatives and *influence* from new magnitudes;
3. Handle the *proportionality* relationship based on these new derivatives;
4. Check for *continuity* from the original magnitudes / derivatives;
5. Clip derivatives at the *extremes* if needed, e.g. don't allow derivative + for magnitude *max*.

Based on these assumptions, we have designed an expected state graph, which can be found in figure 1.

3 Algorithm

This section presents the design decisions made and the underlying logic of the algorithm used. We implement the algorithm using the *Python* language. The *DOT language* [1] and the *PyGraphViz* [1] module have been used to generate the state-graphs and the traces. We have provided a Docker image to make our setup reproducible and have published our source code on Github for peer review.

3.1 Design Decisions

We have used Python 3.7 data classes as the data structure to represent the general qualitative reasoning concepts such as entities, quantities, relationships, and states. These are similar to C++ structs, in that they facilitate optional static typing as compared to more dynamic structures such as Python's dictionaries.

Each quantity space in our bathtub model is represented by enumerations, defining what the possible values the quantity can have. These simplify the way in which conditions can be checked for validity and comparison.

We implement a variety of checks in order to determine the validity of the state-transitions and the states themselves:

Influence check takes a pair of states and computes the derivatives by taking into account the effects of all the direct and proportional influences on the quantities involved in the first state. It then returns a verdict based on whether they match with those in the second state or not.

Magnitude check also takes a pair of states and determines what the magnitudes of the quantities in the second state will be, based on the derivatives of the quantities in the first state. It returns a verdict based on whether they match with those in the second state or not.

Derivative extremity check ensures that the derivative does not increase or decrease when the magnitude of a quantity has reached its maximum or minimum value respectively.

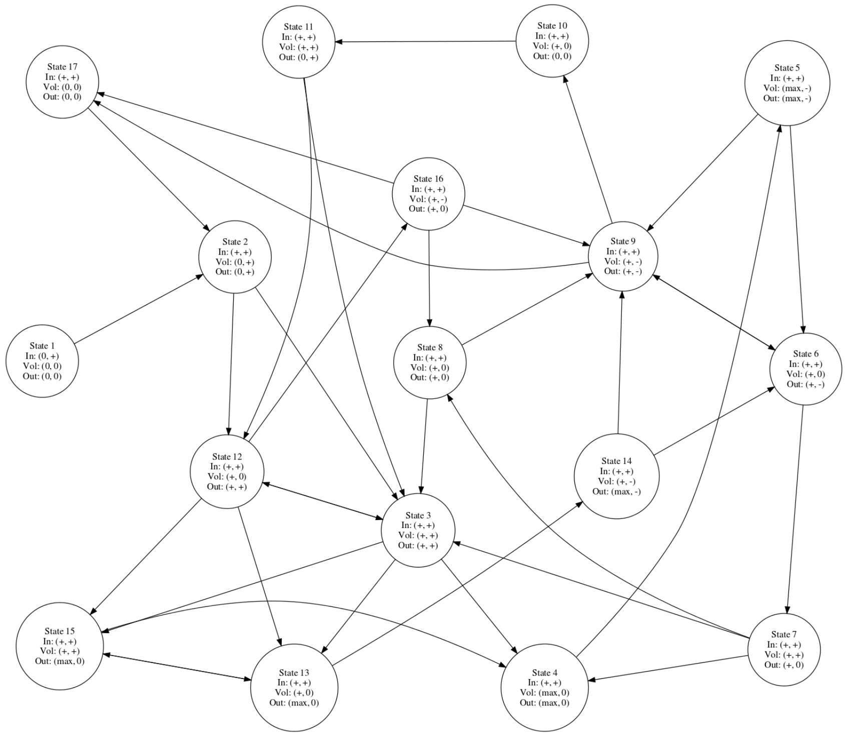
Derivative continuity check ensures that the derivative does not display an abrupt change (skipping a value in its quantity space) as we move from one state to another.

Value correspondence check determines whether the value correspondences hold or not.

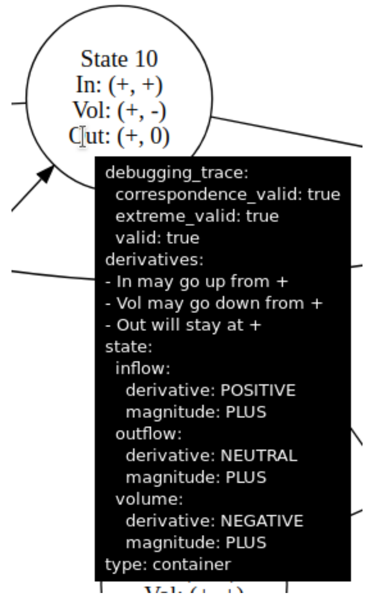
Point range check determines whether the magnitudes in an interval do not change until any at a point have already changed.

Initially, we implemented an algorithm that generated all the states based on all the possible combinations of the magnitudes and derivatives of the quantities involved. It then reduced the size of the state space by discarding the invalid states through indirect / direct influence, value correspondence, derivative extremes and continuity checks.

However, this resulted in some valid states being deemed invalid, as the checks gave contradicting verdicts. Thus, we decided to try a different approach as this method of pruning the states based on the validity checks did not prove effective.



(a) State graph for the basic model



(b) Intra-state trace

The approach that we present in this report utilizes an initial state {Inflow: (0, +); Volume: (0, 0); Outflow: (0, 0)} which models the situation where the water is just about to flow into the sink. Then, we generate the possible states that we can transition to from the initial state. We use the following approach to generate valid states:

1. Determine the quantity magnitudes in the next state(s) from their corresponding derivative directions in the current state.
2. Determine the quantity derivative directions in the next state(s) based on their new magnitudes
   (a) Perform direct influence and determine their new derivative directions.
   (b) Perform indirect influence and determine their new derivative directions.
   (c) Determine their overall derivative directions by resolving the results from the above two sub-steps.
3. Perform a continuity check to ensure whether the derivatives transition smoothly or not.
   - If so, the state is valid, otherwise invalid.
4. Enforce an extremity check to ensure that the derivatives are neutral in case the magnitude reaches a maximum or minimum value.

This process is repeated for each generated state until no more can be generated.

4 Generated State-Graph

The state graph we have generated for the basic design of our model can be found in figure 2a. Unfortunately we had to cut out exogenous and bonus state graphs due to the 5-page limit. While we had also attempted to generate a state graph for the bonus model including exogenous actions, the resulting DOT file turned out at over 2 MB.
Note: The inference of the system behaviour that can be gained from the state-graph(s) and certain statistics that were obtained have been discussed under *Extra Details*.

5 Trace

We have designed both intra- and inter-state traces in our algorithm. We managed to dynamically present these to the user as tool-tips on the nodes and edges, respectively, so that the user would only need to hover over the state or transition (edge) they would like to learn more about.

The information we have integrated in our **intra-state** trace includes:

- Entity type
- Entity state (including quantity magnitudes and derivatives)
- A debugging trace consisting of sanity checks to ensure whether the state is valid, based on:
  - An extremity / clipping check
  - A value correspondence check
- A natural-language summary of the imminent changes to a state's magnitudes, as dictated by its derivatives. This idea could be further expanded on by additionally taking into account influence or proportionality relationships. In our present iteration, for example, one such summary might say:
  - In may go up from +
  - Vol will stay at 0
  - Out will go down from max

Consider the intra-state trace in figure 2b. Through the natural language summary, our trace manages to convey the fact that the magnitude of the outflow may be more than the magnitude of the inflow, thus resulting in a negative derivative for the volume.

The information we have integrated in our **inter-state** trace includes:

- all of the intra-state trace information about the two states covered by the transition (edge)
- A debugging trace consisting of transition-level sanity checks, based on:
  - a check to confirm magnitude changes appear valid based on the derivatives;
  - a check to confirm derivative changes appear valid based on the influence/proportionality relationships;
  - a check to confirm the two states are distinct, as we do not allow duplicate states;
  - a check to confirm the continuity principle has not been violated;
  - a check to confirm the rules around point and interval values are abided in the transition.

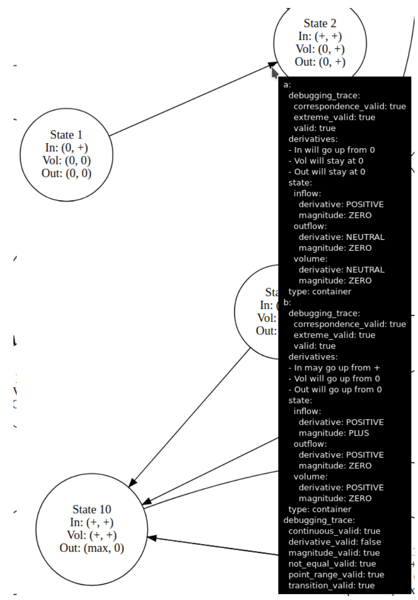


Figure 3: Inter-state trace

Consider the inter-state trace in figure 3. Our inter-state trace has a similar format to that of the intra-state trace. The inter-state trace shows a natural language summary of "In may go up from 0" for the inflow which captures the fact that since the inflow has a positive derivative, it is bound to transition from a point-value to a range-value in the next state. "Vol will stay at 0" and "Out will stay at 0" describe the absence of any change in the magnitudes of volume and outflow in the next state since their derivatives are neutral.

A future version of this trace might expand on this so as to include further information on which rule applications were responsible for which changes among the quantity magnitudes and derivatives between the two states.

6 Extra Details

In the full version of our model, as outlined in figure 1, the outflow is no longer affected directly by the volume, but rather through the height and pressure. The value correspondence between volume and outflow disappears and becomes more explicit through a cascade of value correspondences from volume to outflow through height and pressure.

While in a bi-directional interpretation of the value correspondence rule, one might obtain a very similar result here as for the basic model, we felt like it would be more interesting to see the more full-fledged version of the model in all its glory.

Now, while for the basic model, under our assumptions, we ended up with 17 states and 37 edges, the basic model including exogenous random actions on the Inflow derivative at 66 states and 367 edges, and the full model comprises of 162 states and 555 edges.

This difference stands to reason, as before eliminating impossible states, the basic model would have had $3^5 \times 2 = 486$ potential states (the asymmetry stemming from inflow lacking a max magnitude), while the full model instead has $3^9 \times 2 = 39,366$ potential states. This means that while in the basic model the actual number of states covered 3.5% of the potential space, 13.6% for the exogenous version, while in the full model this was only 0.4%.

While this full network of states is fairly densely connected, at an average of 3.4 rather than 2.1 edges per state, it nevertheless still starts out from a single **source** state ($State1In : (0,+)Vol : (0,0)Out : (0,0)$ for the basic model, $State1\,In : (0,+)Vol : (0,0)Out : (0,0)Pres : (0,0)Hi : (0,0)$ for the full model), although the basic model did not feature any steady end-state given its negative feedback loop, the full model does in fact contain multiple 'final' states, e.g. $State86In : (+,+)Vol : (0,0)Out : (max,-)Pres : (+,-)Hi : (0,0)$.

Although *Inflow* was fixed as a steady positive derivative in our model, *Volume* ended up the next quantity most tending toward the intermediate plus value, at 108 occurrences out of 162 possible states. Likely in part due to the value correspondences, the value distribution for the quantities *Outflow*, *Pressure* and *Height* was a lot more evenly distributed, each getting generally around 70+ occurrences for plus magnitude, with the other half around evenly divided up into the zero and max extremes.

Despite these more intimidating numbers however, as one may see from the causal model graph, the basic feedback loop of the basic model is still intact even in the full model however: although the bonus model spans a few more nodes, the primary cycle in the model contains only a single negative relationship, making for what is essentially the same basic negative feedback loop between Volume and Outflow.

In a soon-to-be published follow-up research, we plan to upgrade our present model using drain plugs and rubber ducks.

In conclusion, we must warn the reader: remember *never* to open the tap to your bathtub — you never know what might happen.

References

[1] Graphviz pocket reference. https://graphs.grevian.org/. Accessed: 2019-4-10.

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

7 Appendix

Quantities Involved	Interpretation
I+(Inflow, Volume)	The amount of inflow increases the volume
I-(Outflow, Volume)	The amount of outflow decreases the volume
P+(Volume, Outflow)	Outflow changes $\propto$ volume changes
VC(Volume(MAX), Outflow(MAX))	Maximum Volume $\implies$ Maximum Outflow
VC(Volume(0), Outflow(0))	Zero Volume $\implies$ Zero Outflow
Bonus Relationships	
P+(Volume, Height)	Volume Changes $\propto$ Height Changes
P+(Height, Pressure)	Height Changes $\propto$ Pressure Changes
P+(Pressure, Outflow)	Pressure Changes $\propto$ Outflow Changes
VC(Volume(MAX), Height(MAX))	Maximum Volume $\implies$ Maximum Height
VC(Volume(0), Height(0))	Zero Volume $\implies$ Zero Height
VC(Height(MAX), Pressure(MAX))	Maximum Height $\implies$ Maximum Pressure
VC(Height(0), Pressure(0))	Zero Height $\implies$ Zero Pressure
VC(Pressure(MAX), Outflow(MAX))	Maximum Pressure $\implies$ Maximum Outflow
VC(Pressure(0), Outflow(0))	Zero Pressure $\implies$ Zero Outflow

Table 1: Relationships between the quantities involved