KNOWLEDGE REPRESENTATION 2019

Qualitative Modelling

applied to a Water Tub System

Luisa Ebner, VUA, 2638040 & Philipp Lintl, UvA/VU, Student nr.2646385

April 12, 2019

Introduction

The demand to represent and model physical systems is ever-present in various scientific domains. However, the development of an adequate model can encounter difficulties in many regards, including the availability of sufficient, numerical information or the predictability of sufficient prior knowledge among the addressed audience. [1] In this context, Qualitative Reasoning (QR) is a modelling formalism, that allows to represent and reason about continuous system processes, especially in situations, where information is vague or incomplete or where due familiarity with mathematical models cannot be presumed. [2]
Qualitative Modelling is based on domain abstraction, whereby continuous system quantities are discretized into a finite number of states, that can be depicted and reasoned with as fixed entities. This enables for a depiction of system processes in a symbolic, human-like manner. System changes can be presented qualitatively and primarily according to the principle of causality. [1]
As to that, QR models aim at providing a broad understanding of a system and its important behavior patterns. One might say, they allow for an intuitive interpretation and qualitative predictions of inherently more complex systems. Especially in the context of scientific education, Qualitative models enable students to reason about the common sense world of system quantities, motions and spaces over time. In the context of professional research, science or engineering, QR can provide useful system overviews prior to more advanced, numerical analyses. [1]

In this report, the principles of Qualitative Modelling and Qualitative Simulation are applied to the simple case of a bathtub system, where water can enter a container through an above placed tab and evacuate though a drain at the bottom of the container.
In section 1, the the system setting is introduced and a corresponding QR model is formulated. Besides the model entities and quantities, the system structure is defined by the relations between, and constraints on the system quantities. Eventually, the causal model of the bathtub system is portrayed graphically.
In section 2, the implementation of an algorithm is described, that automates the reasoning about all possible states and state transitions of the bathtub system. The same generates a qualitative simulation of the bathtub system and creates state graphs of all possible system behaviors according to different scenario options, chosen by the user. Furthermore, it displays an explanatory trace of how consistent behavior paths were created.

1 The Water Tub System

1.1 Model formulation

Model formulation is the first step towards successful Qualitative Reasoning. As to that, the following section is dedicated to the design of a qualitative model for the simple, physical system of a bathtub, that can be filled with water via a tab and emptied via an ever-open drain.

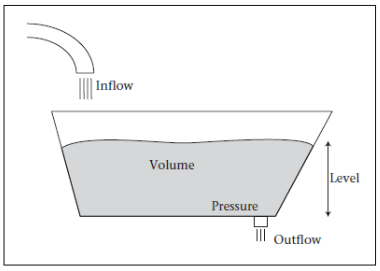

Figure 1: The Bathtub System [2]

1.1.1 Entities & Quantities

In the context of QR, information about system entities and quantities is part of the so-called *model fragments.*

The bathtub system comprises three physical entities. First and foremost, there is a *container* of a finite, unknown size that can be empty or filled with water. Accordingly, we assign the *container* three quantities. If the latter is filled with water, the water column therein can reach up to different *heights* (= level in Figure 1). If the *container* is maximally full, the water height equals the container height. Every *height* value corresponds to a certain *volume* of water in the container and a certain *pressure* of the water column onto the bottom of the container. Clearly, *volume* and *pressure* are maximal at maximal height.

Above the container, there is a *tab*, which has *inflow* as its only quantity. It shall be assumed that the tab can be either closed, open at a constant level, opened increasingly or open at a de- or increasing level.

At the bottom of the container there is a *drain*, which we assign it the quantity *outflow.*

1.1.2 Quantity spaces

In QR, a system quantity is typically defined by the tuple *(magnitude, derivative).* Thereby, the principal idea of QR is domain abstraction and thus the discretization of continuous magnitude and derivative domains into a finite number of ordered symbols representing relevant value/change ranges or interesting, fixed values for comparisons. The transformation of continuous parameters into symbolically representable entities provides a means of abstraction, that regards to the lack of numerical knowledge and allows for a depiction of specifically relevant system behavior.

A QR system generally has a limited domain of validity. In the bathtub system, the magnitude of the quantity *inflow* shall be restricted to the categories '0' and '+'. That is, we discretize the quantity space of water inflow into absent or present. For all other quantities, the space of possible magnitude values shall be discretized to ('0', '+', 'max'). Thereby, '0' and 'max' are landmark values. The landmark '0' is numerically set, whereas 'max' is defined rather symbolically as an unknown, fixed value.

The quantity space used for the derivatives shall be signs for all five quantities. That means, the direction of change is either decreasing ('-'), steady ('0'), or increasing ('+'). This is the minimal choice of abstraction for which continuity assumptions can still be expressed. Indeed, this discretization takes account of all interesting state transition points within the system's nature: Either the container is empty or it is filled with water at a stable, rising or falling level. As to that, it is sufficient to discretize the quantities' continuous value sets by the landmark points '0' and 'max'. In between, the system behavior remains qualitatively equal. [1]

1.1.3 Relations and Constraints

Relationships between system quantities express constraints imposed by the physical world and describe the dynamics of a system. [1] In Qualitative Reasoning, exact functional relations among a set of variables are typically abstracted through the use of qualitative relations. Among these are influences and proportionalities.

In the bathtub system illustrated above, the amount of water *inflow* positively influences the change in *volume* of water in the container. The amount of *outflow* negatively influences the latter. Moreover, there is a propagation of change from *volume* to *height*, from *height* to *pressure* and from *pressure* to *outflow.* Put differently, *height, pressure* and *outflow* are positively proportional to *volume.*

Constraints can explicitly be specified by means of corresponding values, values correspond in the case of an empty and of a maximally filled tub. If the *volume* in the tub is zero, then *height, pressure* and *outflow* need to be zero as well. The same value correspondence holds in case the *volume* has reached its unknown, but fixed, maximum value. Any and all correspondences in the water tub system hold bidirectionally between all quantity pairs (excluding *inflow*).

1.1.4 Additional Assumptions

Additionally, a number of additional assumptions was made to specify the system more distinctively:

1. The drain is an non-closable opening in the container shell. It cannot be closed manually. Whenever there is water in the container, then there is water flowing out the drain.

2. As bathtub has finite (unknown) dimensions, there are fixed, unknown values representing the maximal height, maximal pressure and maximal outflow.

3. The derivatives of *volume, height, pressure* and *outflow* and can directly or indirectly be inferred from the quantity 'inflow'. Only the tap allows external intervention in the

system. Accordingly, we define *inflow* as the only exogenous system quantity.

4. The tub cannot overflow. In a state where the tub is completely full (max volume), the volume can no longer be increasing. Possible derivatives are '0' or '-'. As 'max' volume goes along with 'max' outflow, we assume that in this very moment of the system *outflow* is greater than its *inflow*. All non-exogenous quantities stabilize and then decreases again. Moreover, we assume that *volume* (and thus *height, pressure* and *outflow*) can never reach 'max' magnitude, if it was not 'max' initially. This means, we assume there is some point in the '+' range of inflow, where the outflow becomes greater than inflow. This assumption is reasonable to prevent from practically overflowing tubs, with 'overflowing' as a final state. Finally, it is not sensible to design a tub that can overflow for an infinite amount of time without a change of the system quantities. Additional simplistic assumption would need to be made regarding the containers surrounding.

5. The quantity *inflow* is not given the landmark value 'max'. This is because firstly we are interested in a possible simple model and secondly, irrelevant landmarks can cause undesired branching. [2]

6. There is only a restricted number of possible scenarios for the tab and thus the inflow variable of the system. Our assumption disregards random inflow behavior, as we consider a random scenario neither realistic nor relevant in practice. The set of potential scenarios is rather defined as follows:

   - decreasing: the tab is yet open and gradually gets closed (initial inflow derivative = '-')
   - increasing: the tab is just being opened or yet open and then gradually opened to a growing extent (inflow derivative = '+')
   - stable: the tab remains closed at any time point or is open to a certain, constant extent (inflow derivative = '0')

7. After the system starts, there are no more external interventions to it. There is e.g. no later point in time where the tab is manually closed.

1.2 The Causal Model

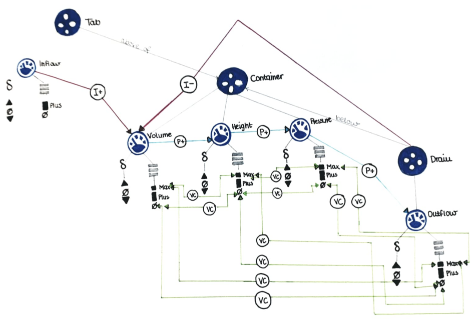

Figure 2: (VC = value correspondence, I+/- = positive/negative influence, P+ = proportionality)

2 Qualitative Simulation

2.1 Implementation

After a successful model formulation and a complete, symbolic representation of the system as a causal model, Qualitative Reasoning is typically concerned with simulating the system's dynamics over time. Even though we do not know the exact values for inflow and outflow or the physical dimensions of the bathtub, a simulation shall enable us to predict the behavior of the bathtub qualitatively. Within QR, Qualitative Simulation (QS) is the reasoning technique to define all possible future behavior categories of a system, forming a coherent behavior path within the system. [3] This complete generation of possible future behaviors is also referred to as envisioning. [1]
In order to automate the capturing of all potential behavior categories for the given bathtub system, we implemented a QS-algorithm. The same shall not only derive all possible behavior paths, but also output a respective graph and a trace of the taken implementation steps. The concrete implementation process that shall be described in detail hereafter.
First of all, we decided to employ the highly flexible, object-oriented programming language Python to implement a qualitative simulation of our given bathtub system. Not only does Python enable for an elegant, straightforward combination of lists, classes and objects, it also commands the library `graphviz`, which is eventually used to create state graphs of coherent system behaviors.
We start off defining the discretized quantity spaces and landmark values as lists. As the system representation is symbolic, we decide to represent the elements of the discretized quantity spaces as strings. Full states of the sys-

tem are represented as objects of the class *Entity* with the list of *quantities* as its property, e.g. `state x = State([inflow, volume, height, pressure, outflow])`.

The quantities themselves of the system are implemented as objects of the class *Quantity*, which receive a name, a quantity space, a current magnitude, a current derivative, an affects-list, and a correspondence list as parameters. Given a closed tab, the quantity *inflow* is e.g. represented as

Quantity('inflow', ['0', '+'], '0', '0', ['I+', 'volume'], [])

After having defined all system variables, the function `itertools.product` is used to create a list of all possible combinations of *(magnitude, derivative)*-tuples for all five system quantities. Based thereon, a list of 39366 state objects is created. It shall be noted that this list contains the complete set of valid as well as invalid states.

As to that, the next implementation step is to remove all invalid states. This is done by checking for landmark correspondences, influence and proportionality relations within the state's quantities. Moreover, we examine two additional constraints assumed for the system. Firstly, if there is no more inflow at a time point where the bathtub is maximally filled, then the derivatives of all non-exogenous quantities have to be '-'. Secondly, if the volume and thus all other non-exogenous system quantities have the magnitude 'max', then the accompanying derivative can no longer be '+'. All together, we check for *intra-state consistency* and remove all states that are internally inconsistent. Thereafter, merely 24 valid states remain as a basis to the simulation of possible system behaviors.
According to the fifth assumption in section 1.1.4, we restrict the exogenous quantity *inflow* (and thus the entire bathtub system) to a limited set of reasonable scenarios. Consequently, we allow the user to select one of these three scenarios via keyboard input. Accepted inputs, specifying the system's inflow derivative, are 'i' for increasing, 'd' for decreasing and 's' for stable. In accordance therewith, a state subset is selected out of the 24 states at hand. The subset for 'i' consists e.g. out of all states with a '+' at the inflow derivative. This very subset (9 states) will later on be used to create a state graph for possible 'increasing' system behavior paths.
Now that we are given an implementation of the states themselves, possible state transitions or neighbor states are specified by the algorithm. Transitions are defined as the time points at which the qualitative state of the system changes. This means, at least one variable or derivative reaches or leaves a landmark of its quantity space. Transitions either apply from one time point to the time interval starting at this time point or from one time interval to its ending time point. [2]. The former are called instantaneous transitions. The latter those that have positive durations.
Due to the opposed influences on *volume*, predictions on system derivatives are not unique. [1] In other words, successor states cannot always be determined unambiguously. Sometimes there is a number of possible transitions and thus neighbors. Consequently, our QS-algorithm finds a list of all values, that can transition in the next step, given a certain current state. If changes are not instantaneous, the algorithm creates a list with all possible neighbor states. Otherwise, the instantaneous transition from a landmark value to a value range is applied. Based on the possible new states that are associated to one of the possible start states, the algorithm checks which states of all the valid states can be following. Particularly, if instantenous changes appear, the only possible new state is the one with all of the changes applied. If no instantenous changes appear, we add a list of all of the possible combinations to the list of new states. For both cases the list of new state is used to determine the possibly following states, defining one behaviour path. Repeating this procedure for each possible start state, yields a complete adjacency list, that is then used to create the respective graphs.

Additionally, a tracefile `trace.txt` was created by printing the basic algorithm steps, as well as one exemplary state description and one inter state transition.

2.2 State Graphs

Our QS-algorithm defines a set of potential, initial states and repeatedly generates a sequence of chronologically ordered successor states. Because we did not set one initial state and neighbor states cannot generally be determined uniquely, the simulation branches at every possibility. Eventually, the algorithm's state lists shall be depicted graphically as a state graph in tree form, where every state is represented as a node with the variables' qualitative values and directed arrows represent the transitions between states. Finally, one coherent behavior is a path, that goes from the root of the tree to a leaf node. The latter is obtained when a state is a so-called no-change state (= a state, where the successor state would be identical). [2]

As explained in chapter 1.1.4, we roughly distinguish three scenarios of possible bathtub behaviors depending on the exogenous variable *inflow*. Depending on the user input, we result in three different state graphs.

1. If the decreasing scenario is selected,the state graph may be described as follows: Starting from one common root node, the graph branches six states with a negative inflow derivative as potential, initial states. In either case, the initial tuple for inflow is ('+', '-'). The other initial quantity tuples express that the tub is either full at a constant level, full but already decreasing or filled to some extent while being stable or decreasing; other than that, the tub can be initially empty, with a positive derivative, then fill up and stabilize at some water level before decreasing as well. In the end, we depict the scenario of a bathtub, where the water decreases from the container due to a primarily decreasing level of inflow that eventually turns zero.

2. In the stable scenario, the entire subset of states with a stable inflow derivative can be initial states of a behavior path. The first stable inflow option is to keep the tab closed at all times. Corresponding scenarios are the empty tub that remains empty forever an the initially full tub, where water is gradually evacuating. Alternatively, there can a positive, stable inflow. Then again, it depends on the initial state of the container. One initial state shows e.g. the empty tub, with a positive derivative due to the positive inflow magnitude. Primarily, all non-exogenous quantities transition to ('+','+'). At a certain point in the positive volume - height - pressure and outflow value range, there is equally much water flowing into and out of the container. At this very time point, the water level stabilizes to ('+', '0'). As the exact amounts of water in- and outflow are unknown, the water level can stabilize (infinitely weaver between ('+', '+') and ('+', '0') or gradually decrease down to the bottom of the container.

3. The 'increasing' state graph initially branches 9 states with positive inflow derivatives. We imagine someone opening a closed or yet open tab more and more. (('0', '+') or ('+','+')). Once more, the container can either be empty, filled to some or the maximal extent. The graph models tub that is throughout decreasing despite an increasing inflow level. Furthermore, it shows the initially empty tub, that fills up, stabilizes at a positive or the maximal level, and sinks only thereafter. Finally, the graph models a system behavior where the water level stabilized somewhere in the '+' region.

2.3 Discussion

Summing it up, our simulation results conform with decent behavior of a simple bathtub system. We left possibly much freedom in our assumptions about the initial state of the system. All valid states function as initial states in one of our three scenario options. We deem this a favourable feature of our state graphs, as users can select any potential start state right away. Fixing the initial state was deemed a needless restriction to the system behaviors. For sake of clarity, we assigned the 24 valid states into three inflow categories and generated state graphs for every subset. We regard our three categories as particularly reasonable in practice: A user of the bathtub will either open a closed or slightly open tab at an increasing level, or he will close an open tab more and more until it is completely closed again. If there is no direct manual intervention, the tub can either be closed or open. Our assumption, that the outflow will at some point exceed the inflow, leads us to one common no-change state for all 3 scenarios. We can never have an empty bathtub that gets filled up to its maximal level and no overflowing state. However, these are both unfavourable system states in practice. A good system should prevent them from occuring. Finally, we defined the drain as non-closable. In reality however, bathtubs often have a drain that can manually be closed by the user. As to that, it might certainly be interesting to model *outflow* as a second exogenous variable. At the same time, this makes the system behaviour distinctively more complicated and corresponding state graphs more difficult to understand.

References

[1] Kenneth D Forbus. Qualitative modeling. *Foundations of Artificial Intelligence*, 3:361–393, 2008.

[2] Louise et al. Travé-Massuyès. Mathematical foundations of qualitative reasoning. *AI magazine*, 24(4):91–91, 2003.

[3] Mehmet Fatih Hocaoğlu. Qualitative reasoning for quantitative simulation. *Modelling and Simulation in Engineering*, 2018, 2018.