Knowledge Representation Project 2
Qualitative Reasoning

Wolf, Florian - 12393339 (UvA)
flocwolf@gmail.com

July 11, 2019

1 Introduction

This report presents a Qualitative Reasoning (QR) project on to a real world-problem as part of the course *Knowledge Representation* by Professor Frank van Harmelen at Vrije Universiteit Amsterdam.

QR is a part of Artificial Intelligence (AI), it represents continuous systems with discrete symbols. Continuous physical values are simplified into quantity space [1]. In this project we consider a container such as a sink or a bath tub. This container has an exogenous water inflow such as a tab and a drain for water outflow. The goal is to create an algorithm which generates all possible sates and their transitions between each other. These transitions will be represented as connections with in a graph structure. Finally the results will be analyzed and discussed.

2 Design

We deigned our QR model to represent the real-world behaviour. The sink is influenced by three different quantities. There are inflow, volume and outflow. Each quantity is represented by a value and a gradient. The value has three discrete states in quantity space 1. An exception is the inflow which does not have a maximum. The gradient is represented in three states as well of which one is negative 2.

$$[0, +, max] \tag{1}$$

$$[-, 0, +] \tag{2}$$

These representations are common knowledge adopted from qualitative reasoning practices. With this design rules we can go a head and make assumptions to constrain our model.

3 Assumptions

Before creating a model, assumptions have to be made to constrain and simplify the real-world physics. With these assumptions we can create all possible states and define the state transitions between them. The first part of the assumptions will determine which states are physically possible and which are not. In 1 we present the assumptions on the relationships between quantities.

Quantity	Magnitude	Q. Space	Quantity	Magnitude	Q. Space
Volume	Value	same	Outflow	Value	same
Volume	Gradient	same	Outflow	Gradient	same
Volume	Value	max	Volume	Gradient	0 or -
Outflow	Value	max	Outflow	Gradient	0 or -
Volume	Value	0	Volume	Gradient	+ or 0
Outflow	Value	0	Outflow	Gradient	+ or 0
Inflow	Value	0	Inflow	Gradient	+ or 0
Inflow	Value	+	Volume	Value	+ or max
Inflow	Value	0	Volume	Gradient	0 or -
Volume	Value	max	Inflow	Gradient	0 or -
Inflow	Value	+	Volume	Value	+ or max
Inflow	Gradient	+	Volume	Gradient	+

Table 1: Assumptions of how the first entity constrains the second entity.

Further assumptions are that the sink has a maximum of holding water. When this maximum is reached the gradient of the inflow can not be positive. The outflow is directly related to the water volume in the sink. The value as well as the gradient of volume and outflow are always the same. When the Inflow is zero, the gradient of the volume can not be positive.

The assumptions about the connections between states are fewer and simpler. The gradient of the first state determines the value of the next state. If this applies for all three entities we assume a possible state-transition. In practice this means that if the gradient is zero, the value should stay the same. If the gradient is positive the next value should rise and the other way around for a negative gradient.

These assumptions define our model and are the base of the following algorithm. The representation is a trade-off between complexity and correctness. The more assumptions and constrains we choose, the closer we can model the real-world but with the cost of rising complexity.

4 Algorithm

The algorithm is divided in three steps. The first step creates all states. In the second state we filter those steps according to our assumptions and in the last step we create a graph for visual representation.

To create all possible states we loop over each quantity in the defined quantity space. In total we create 486 different states. Each state is saved as a list of three tuples. There is one tuple for each magnitude and each tuple contains the value and gradient.

These states are then filtered by applying the constrains defined in 4. We loop over all states and initiate each state with a `True` flag. If the state fails to compile with one of the constrains, the flag is set to `False`. Only states which satisfy all constrains are passed on. From the initial states only 15 remain.

To define the transitions between the states we index the states with individual IDs. We loop over all possible pairs of states and check the defined connection constrains. An exception is made for the quantity inflow, which does not have a maximum and therefore can have a positive gradient without changing the positive value in the next transition. In total 44 connections were established. The connections are unilateral directed.

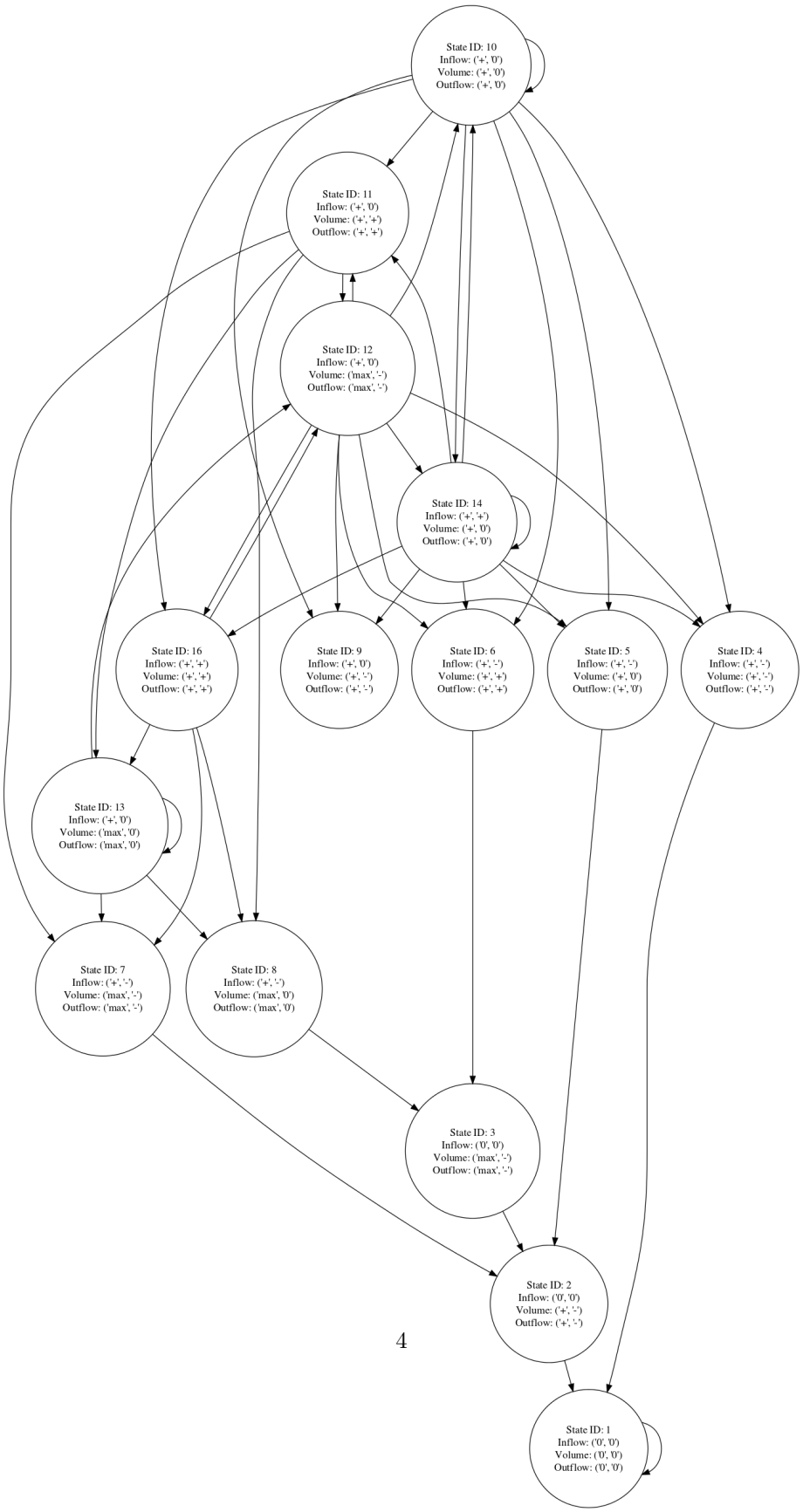
In the last step of the algorithm we create a graph structure using the python library `pygraphviz` [2]. This library represents graphs by connecting nodes trough edges. For each state we create a node and use the established connections to draw edges between the nodes. The graph structure is saved as `png` file and can be reviewed in 5

The code was written in Python 3.7.3 on Linux. It can be found on GitHub as public repository [3].

5 Results

After running the code, we got a graph structure representing all possible states and state-transitions. These are based on the assumptions made in 4.

In 1 state 10 can be considered the initial state, as reference to when the tab is open and water steadily flows into the sink and out via the drain. This state can be reached from other states and is therefore not solely initial. State 1 is the terminal state, representing the tab being turned off and all the water drained from the sink. The constrains of our previously made assumptions prevent this state from transitioning to any further states. The states 2, 3, 4 and 7 are pre-terminal and lead solely to the final state. State 10, 13 and 14 are loop-states which can stay repeatedly transition to the same state.


State ID: 10
Inflow: ('+', '0')
Volume: ('+', '0')
Outflow: ('+', '0')
State ID: 11
Inflow: ('+', '0')
Volume: ('+', '+')
Outflow: ('+', '+')
State ID: 12
Inflow: ('+', '0')
Volume: ('max', '-')
Outflow: ('max', '-')
State ID: 14
Inflow: ('+', '+')
Volume: ('+', '0')
Outflow: ('+', '0')
State ID: 16
Inflow: ('+', '+')
Volume: ('+', '+')
Outflow: ('+', '+')
State ID: 9
Inflow: ('+', '0')
Volume: ('+', '-')
Outflow: ('+', '-')
State ID: 6
Inflow: ('+', '-')
Volume: ('+', '+')
Outflow: ('+', '+')
State ID: 5
Inflow: ('+', '-')
Volume: ('+', '0')
Outflow: ('+', '0')
State ID: 4
Inflow: ('+', '-')
Volume: ('+', '-')
Outflow: ('+', '-')
State ID: 13
Inflow: ('+', '0')
Volume: ('max', '0')
Outflow: ('max', '0')
State ID: 7
Inflow: ('+', '-')
Volume: ('max', '-')
Outflow: ('max', '-')
State ID: 8
Inflow: ('+', '-')
Volume: ('max', '0')
Outflow: ('max', '0')
State ID: 3
Inflow: ('0', '0')
Volume: ('max', '-')
Outflow: ('max', '-')
State ID: 2
Inflow: ('0', '0')
Volume: ('+', '-')
Outflow: ('+', '-')
State ID: 1
Inflow: ('0', '0')
Volume: ('0', '0')
Outflow: ('0', '0')

6 Conclusion

The model successfully returns all possible states and their connections. It can be used to represent real-world entities considering the assumptions made in 3. Correctness can be argued especially in regard to the exogenous influence. The tab being turned off would not be the only terminal state we could think of. Further the proportionality between inflow and outflow magnitude is not clearly defined. The maximum outflow could possibly be higher than the inflow what would lead to the case in which the water volume in the sink never reaches its maximum. As mentioned before QR is a trade-off between correctness and complexity. This project could be extended by implementing the laws of physics such as pressure depending on the water volume height or fluid dynamics impacting the in and outflow by velocity, temperature and fluid-friction. Trough out the process of QR modeling a key aspect is to keep consistency and avoid contradictions [4].

Regarding that the goal of this project was rather to familiarize with the concepts of QR and creating an automated representation of a real-world dynamic system we are satisfied with the result and look forward to dive deeper into the study of *Knowledge Representation*.

References

[1] Daniel G Bobrow. *Qualitative reasoning about physical systems*. Elsevier, 2012.

[2] pygraphviz. Documentation. *GitHub*, 2018.

[3] 3lLobo. Qualitativereasoning. *GitHub*, 2019.

[4] Zhisheng Huang, Frank Van Harmelen, and Annette Ten Teije. Reasoning with inconsistent ontologies. In *IJCAI*, volume 5, pages 254–259, 2005.