The Dam Case Study

Henning Dierks[1]

Hochschule für Angewandte Wissenschaften Hamburg

In this case study we consider a dam that impounds water and uses this to produce energy with the help of turbines which the water has to pass. We assume a steady inflow of water towards the dam. Relevant for safety is the water level. To control this level the dam is equipped with a number N of turbines which can be controlled separately.

Turbines

These turbines have three modes. In mode *low* a turbine consumes a reduced amount of water per time unit. This is the preferable mode because it is the most efficient one in terms of energy produced per water unit. In mode *high* a turbine consumes the maximum amount of water it can consume. The mode *maintenance* represents a turbine that is stopped for maintenance reasons. Here, no water passes the turbine. The turbines have to switch into the maintenance mode after a given number (*threshold*) number of changes from the *low* to the *high* mode. It takes a given amount of time (*duration*) until maintenance is finished. To keep track of this each turbine needs a maintenance counter. A hybrid automaton describing the behaviour of the turbine is given in Fig. 1.

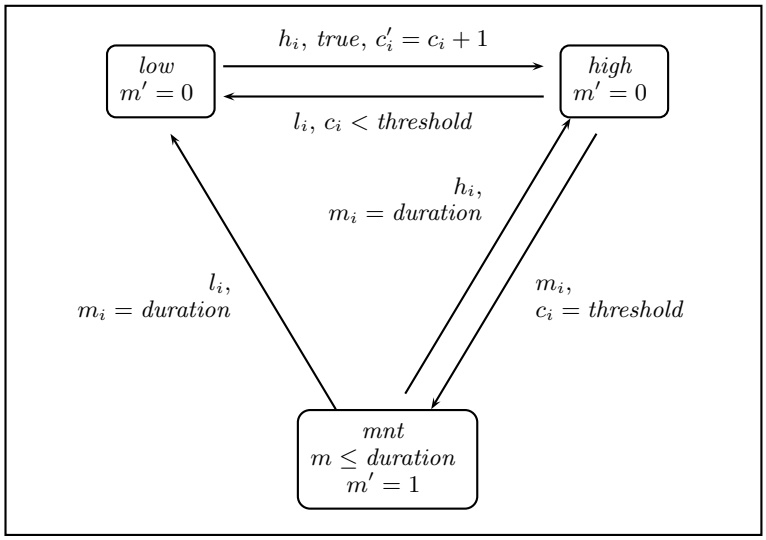


Fig. 1. Turbine i as hybrid automaton

Water level

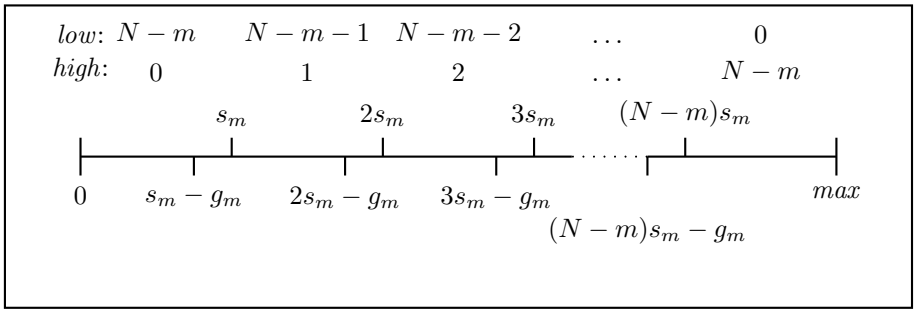
The water level of the dam is a continuous variable which is influenced by the following factors:

- Incoming water increases the level but we only have both upper and lower bounds *max_in* and *min_in* for the rate.
- Turbines in *low* mode decrease the water level with rate *lr* ("low rate"). If a turbine is in *high* mode it decreases the water level with rate *hr* ("high rate"). Otherwise, ie. in the *maintenance* mode, the rate is 0.

The safety property of this system is that the water level shall always stay within given bounds (*min* and *max*). Wlog. we set $min = 0$.

Controller

The controller of the system can observe the current water level and may change the modes of the turbines. In our model we consider a controller with the following strategy. Depending on the current status of the turbines, say m are in *maintenance* mode, the controller partitions the interval $[0, max]$ into $N+1-m$ many parts:



The size s_m of the partition is $\frac{max}{N-m+1}$. If the water level raises and reaches the border $k \cdot s_m$ the controller switches one turbine from mode *low* to *high*. Here, it always selects the turbine in *low* mode with the lowest index. If the water level sinks and reaches the border $k \cdot s_m - g_m$, the controller determines which turbine is in *high* mode and has the highest index. This turbine is then switched to the *low* or *maintenance* mode. It depends on the status of the turbine's counter in which mode it will switch. The purpose of $-g_m$ in the term above is to ensure that the controller remains stable for a fixed duration. We set $g_m = \frac{1}{4}s_m$. A hybrid automaton describing the behaviour of the controller for 2 turbines is given in Fig. 2 on a torus[1]. Note that in this figure we use F as abbreviation for $[min\_in, max\_in]$, ie. the range of the inflow.

[1] This shall improve readability. Arrows that hit the border are continued on the opposite border

Initial state and

Initially, all turbines are in mode *low*, the water level is in $]0, s_0[$, and all maintenance counters are 0. The controller is initially in state $low - \cdots - low$.

Given concrete instances of all parameters the verification question is whether the controller is able to keep the water level always within the given bounds.

Parameters

Here is an overview of all parameters and their meanings in the case study:

Parameter	Description
max	The maximum water level. The system shall never reach this water level
min	The minimum water level. The system shall never reach this water level. It is set to 0.
min_in	The minimum rate of inflowing water
max_in	The maximum rate of inflowing water
lr	("low rate") With this rate the water level sinks if a turbine is working mode *low*.
hr	("high rate") With this rate the water level sinks if a turbine is working mode *high*.
duration	The time that a turbine spends in the maintenance mode
threshold	The number of switches from mode *low* to *high* that cause the necessity to have a *maintenance* mode after the *high* mode.

Modelling in Phaver

To construct a model for Phaver that describes this case study is basically straightforward as Phaver's syntax entails (almost) all syntactical features needed for hybrid automata. Hence, we will not go into details here because it does not provide any surprising insights. However, there is one aspect that is important for Phaver's performance when it verifies an instance of the dam case study. The only missing syntactical feature of Phaver are discrete variables. The turbines need a counter of type integer with bounded range but this is not directly expressible in Phaver's syntax. Hence, we have two options to model them:

1. *Encoding in the discrete state space*: We could encode the value of the turbine's counter into the discrete state space, ie. we increase the number of locations appropriately. That means that the number of locations of the turbine's hybrid automaton grows from 3 to $3 \cdot (threshold + 1)$ for *each* turbine. This increases the discrete state space by the factor $(threshold + 1)^N$.
2. *Encoding in the continuous state space*: We could encode the integer variables by Phaver's real valued variables. Then we have to set the derivative to 0 in all locations. When a discrete transition is fired we can change the value as needed.

Both alternatives have advantages and disadvantages. An encoding using the discrete state space has the drawback that Phaver does not have efficient data structures to handle large discrete state space symbolically. On the other hand, the encoding using continuous variables makes all computations on the continuous data structures slower.

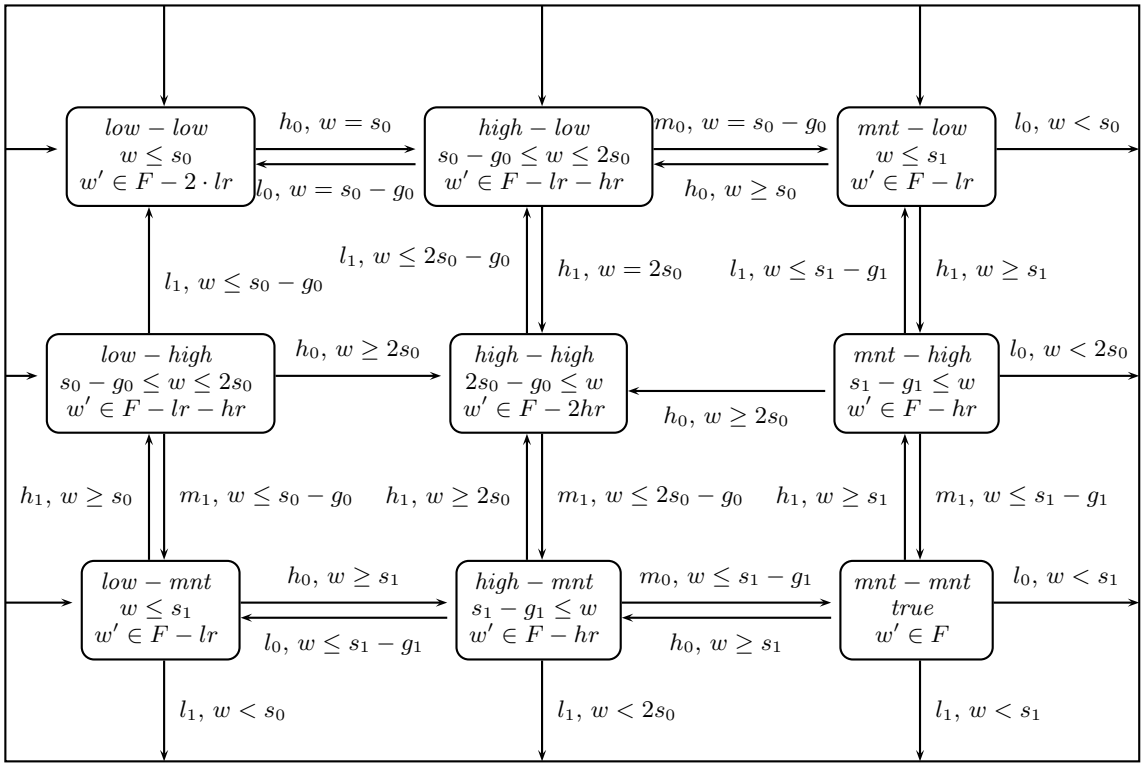


Fig. 2. Controller for 2 turbines on a torus