Nonlinear Analysis: Hybrid Systems 53 (2024) 101504

Contents lists available at ScienceDirect

Nonlinear Analysis: Hybrid Systems

journal homepage: www.elsevier.com/locate/nahs

Optimal control strategies for stormwater detention ponds☆

Martijn A. Goorden [a,*], Kim G. Larsen [a], Jesper E. Nielsen [b], Thomas D. Nielsen [a], Weizhu Qian [a], Michael R. Rasmussen [b], Jiří Srba [a], Guohan Zhao [b]

[a] *Department of Computer Science, Aalborg University, Aalborg, 9000, Denmark*
[b] *Department of the Built Environment, Aalborg University, Aalborg, 9000, Denmark*

ARTICLE INFO

Dataset link: https://doi.org/10.5281/zenodo.6592380

Keywords:
Stochastic hybrid systems
Switching control
Safe and optimal control
Hydroinformatics
Stormwater detention ponds

ABSTRACT

Stormwater detention ponds are essential stormwater management solutions that regulate the urban catchment discharge towards streams. Their purposes are to reduce the hydraulic load to avoid stream erosion, as well as to minimize the degradation of the natural waterbody by direct discharge of pollutants. Currently, static controllers are widely implemented for detention pond outflow regulation in engineering practice, i.e., the outflow discharge is capped at a fixed value. Such a passive discharge setting fails to exploit the full potential of the overall water system, hence further improvements are needed. We apply formal methods to synthesize (i.e., derive automatically) optimal active controllers. We model the stormwater detention pond, including the urban catchment area and the rain forecasts with its uncertainty, as hybrid Markov decision processes. Subsequently, we use the tool Uppaal Stratego to synthesize using Q-learning a control strategy maximizing the retention time for pollutant sedimentation (optimality) while also minimizing the duration of emergency overflow in the detention pond (safety). These strategies are synthesized for both an off-line and on-line settings. Simulation results for an existing pond show that Uppaal Stratego can learn optimal strategies that significantly reduce emergency overflows. For off-line controllers, a scenario with low rain periods shows a 26% improvement of pollutant sedimentation with respect to static control, and a scenario with high rain periods shows a reduction of overflow probability of 10%–19% for static control to lower than 5%, while pollutant sedimentation has only declined by 7% compared to static-control. For on-line controllers, one scenario with heavy rain shows a 95% overflow duration reduction and a 29% pollutant sedimentation improvement compared to static control.

1. Introduction

Urbanization and climate change poses two major challenges related to urban stormwater management: flooding of the urban area, and the environmental impact on receiving waters from urban catchment runoff [1,2]. A storm drainage system collects and conveys the runoff generated from impervious urban surface areas, like roads and other man-made surfaces, and discharges the excessive volume into receiving waterbodies in order to avoid urban flooding. The urban runoff carries high concentration of pollutants from washoff of the urban surfaces [3]. Stormwater detention ponds are the most commonly used stormwater management tool for avoiding or reducing the impacts of stormwater discharges [4]. Negative impacts of stormwater direct discharge include unnatural disturbances in sediment transportation, environmental pollution, and downstream flooding [5,6].

☆ This research is partially supported from the Villum Synergy project CLAIRE and the ERC Advanced Grant LASSO.

* Corresponding author.

E-mail addresses: mgoorden@cs.aau.dk (M.A. Goorden), kgl@cs.aau.dk (K.G. Larsen), jen@build.aau.dk (J.E. Nielsen), tdn@cs.aau.dk (T.D. Nielsen), wqian@cs.aau.dk (W. Qian), mrr@build.aau.dk (M.R. Rasmussen), srba@cs.aau.dk (J. Srba), guohanz@build.aau.dk (G. Zhao).

https://doi.org/10.1016/j.nahs.2024.101504
Received 30 May 2022; Received in revised form 13 March 2024; Accepted 4 May 2024
Available online 10 May 2024

Urbanization and climate change have significantly increased the costs of constructing, maintaining, and upgrading detention ponds for water utility companies to ensure efficient and safe stormwater discharge. For example, the Danish Water and Wastewater Association estimates that the cost of updating the Danish storm water systems will be between 0.6 and 1.3 billion euros [7]. The discharge permits have been increasingly changed over the last 15 years, so even recently constructed stormwater detention ponds do not comply with present recommendations (see [8]).

The requirements for the design of stormwater detention ponds are, in general, based on the maximum allowed discharge flow rate into the nearby receiving stream or river, the probability of emergency overflow, and the concentration of pollutants in the discharged water [9]. In Europe, these requirements are derived from the European Water Framework Directive [10] and discharge permits issued by the local authorities.

Currently, the most common discharge strategies involve static settings to control flows (when there is stormwater in the detention pond) into the stream without taking into account the actual capacity of the receiving stream. These strategies do not necessarily comply with the regulations set by the European Water Framework Directive, as pointed out by [11]. The use of active controllers that change the outflow dynamically, also called real-time controllers or "smart" stormwater management [12], has the potential to improve the efficiency of stormwater infrastructure.

Existing work on the design of active control strategies for urban water systems primarily focus on sewer systems and wastewater treatment plants, which include detention ponds as subsystems, see for example [11–16]. Controlled discharge from detention ponds has also been studied. For example, in [17], a real-time controller is designed for improving the efficiency of particle removal. Further improvements are presented in [18,19], where off-line strategies take weather forecasts into account. However, these works focus primarily on particle removal efficiency as the objective, as most of the pollution is bound strongly to organic and inorganic particles. Furthermore, the underlying rule-based control strategies are manually derived and the discharge output can only be changed infrequently, like once a day.

More recently, real-time controllers that prevent overflow of ponds while also maximizing the retention time were presented in [20,21]. Here, a controller active during wet periods is obtained by solving a linear programming problem, yet during dry periods a manually designed rule-based controller is used. Similarly, [22,23] present real-time controllers for rainwater harvesting systems, which are single-household tanks that collect rainwater for domestic water demands like flushing toilets. While in [22] four different controllers were designed manually, in [23] the authors used the evolutionary algorithm NSGA-II [24] to obtain a safe and optimal controller. In [25] deep reinforcement learning is used to obtain a system-level controller for a coastal urban stormwater system. Even though all these works mention that weather forecasts are inherently stochastic, none of the optimization problems take stochasticity explicitly into account. In our work, we show how stochastic weather predictions can be incorporated into the model.

In the field of cyber–physical systems, there is considerable research in deploying formal methods for verifying or even synthesizing controllers for stochastic hybrid systems. For example, the tool Uppaal Stratego [26] is able to synthesize safe and near-optimal controllers by combining model checking and reinforcement learning. Case studies using Uppaal Stratego include battery aware scheduling problems [27], adaptive cruise control [28], floor heating [29], and fleet management for autonomous robots [30].

The objective of this paper is to deploy reinforcement learning, in particular Q-learning [31], to automatically synthesize active control strategies for stormwater detention ponds, where we maximize particle sedimentation residual time while also minimizing the (expected) emergency overflow duration. As basis for the control synthesis, we model the stormwater detention pond as a hybrid Markov decision process utilizing a combination of differential equations and (stochastic) timed automata. By using this modeling formalism, this paper extends existing work by explicitly model uncertainties inherent to weather forecasts, which controller synthesis then can take into account. The synthesized controller is able to periodically change the discharge flow, thus allowing a more rapid and precise response to uncertain weather events. We synthesize both off-line and on-line controllers, with the on-line controllers using model-predictive control [32]. Simulation experiments of a real-world detention pond are used to compare the performance of synthesized active control strategies with the performance of the current static control.

This paper is an extended version of the conference paper [33]. First, we include the formal definitions of the used mathematical modeling framework. Second, we describe the full model in this paper, where the conference paper only described the main components. Third, model-predictive control has been added where new strategies are synthesized on-line. For this, we combined Uppaal Stratego, for synthesizing the controllers, with EPA-SWMM [34], a domain specific stormwater simulator. Fourth, we include additional simulation experiments for off-line control. Finally, we discuss what the synthesized strategies can and cannot guarantee, especially with respect to safety (in our case emergency overflow). This is reflected mainly in the control problem formulation and the results discussion.

This paper is structured as follows. Section 2 introduces stormwater detention ponds in more details and Section 3 introduces the preliminaries necessary for this paper. The full model and the control problem formulation are presented in Section 4. The main results for off-line controller synthesis are presented in Section 5 and the main results for on-line controller synthesis in Section 6. Section 7 concludes the paper.

2. Storm water detention ponds

Stormwater detention ponds collect stormwater from urban build-up areas, like streets, roofs and parking lots. When it is raining or snow is melting, two main risks arise. First, the urban discharges can exceed the capacity of nearby streams. Second, urban pollutants conveyed by the stormwater can degrade the ecosystem of the receiving waterbodies. Stormwater detention ponds aim to mitigate the impact of both risks. Stormwater detention ponds can be characterized by being either a wet or a dry detention

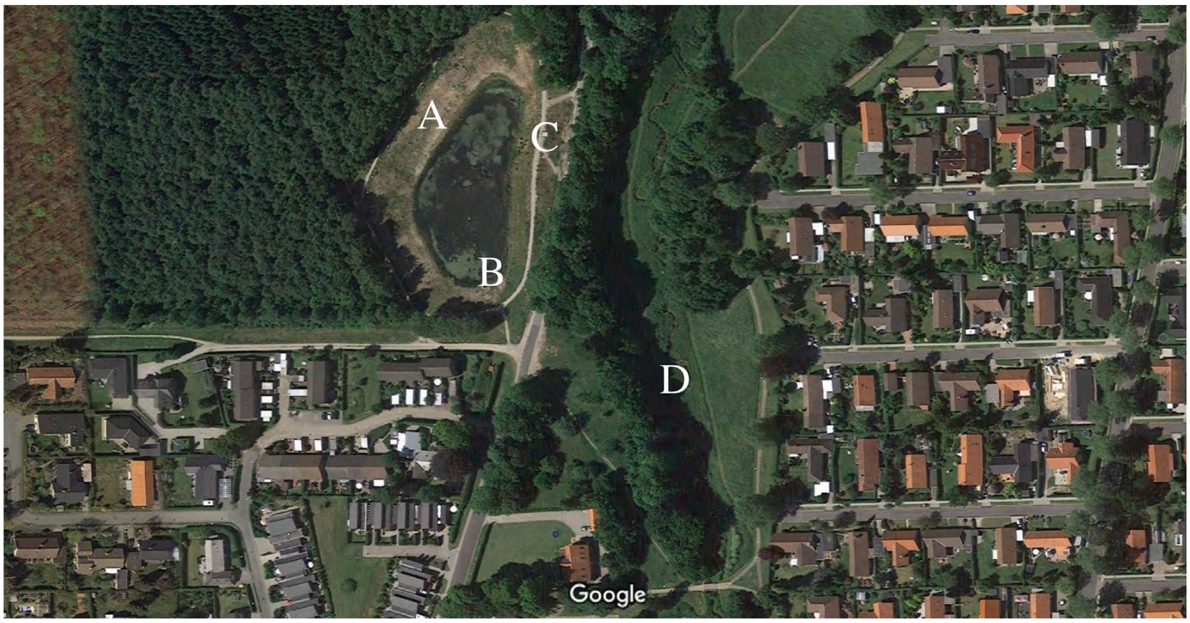


Fig. 1. Satellite image of the Vilhelmsborg Skov pond south of Aarhus, Denmark. A is the detention pond, where the inlet is located at B and the outlet at C. Stormwater is discharged in the stream labeled with D, which runs from the south to the north.
Source: Image from Google Maps.

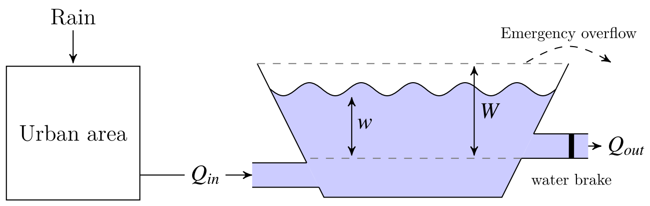


Fig. 2. Overview of the storm water detention pond.

pond. A wet detention pond always has a minimal amount of constant water in it, while a dry one can empty completely (hence the names). In our case study, we focus on a wet detention pond.

An example of such a pond is shown in Fig. 1. The satellite image shows the Vilhelmsborg Skov pond south of Aarhus, Denmark. Labeled by A is the storm water detention pond, partially filled with water. It collects the water from the neighborhoods south of it through the sewer system next to the roads, entering the pond through B. The pond's outlet is indicated by C, which connects to the stream labeled with D. This stream runs from south to north and discharges the water from the neighborhood.

Currently, stormwater detention ponds are often designed with static outlet flow regulator creating a capped outlet flow into the stream. The capacity of the stream (reflected in the issued permits) dictates the maximum outlet flow of the pond. This capped outlet flow is determined by criteria such as the impervious area of the connecting urban catchment and the predicted return period of emergency overflows. This in turn determines the pond volume.

Recent research has focused on the design of energy efficient dynamic outlet flow regulator.[1] Having these flow regulators allows utility companies to incorporate active (or real-time) control into their design of the stormwater detention ponds. This has the potential to enable more efficient detention pond designs and reduce the negative effect of the two aforementioned main risks.

Fig. 2 shows a schematic overview of the wet stormwater detention pond. Rain water falls into an urban area, like a neighborhood, university campus area, or a highway, and is transported to a nearby pond via under sewer system and overland flows. Rain water enters the pond through inlet Q_{in} and exits it through outlet Q_{out} into a nearby receiving waterbody. A water brake in the outlet limits the outflow. Due to the positioning of the outlet pipe, there is a permanent water level in the pond (indicated with the lower horizontal dashed line in the figure). The variation of the water level above this permanent water level is indicated by w. When the maximum water level, indicated by W, is reached, there will be an emergency overflow from the pond.

[1] See project webpage at https://www.danva.dk/viden/vudp/projektuddelinger/relevand/ (in Danish).

We consider the evaporation and infiltration/exfiltration processes to be minor, thus excluding their representation from our model. At the same time, water leakage is assumed to be case specific with relative low likelihood, and will thus be neglected for our general cases. It is shown in [35] that these water flows are negligible compared to the storm water flow. Our control problem is to synthesize a controller that maximizes particle sedimentation (by having a large amount of water in the pond) while also minimizes (the probability of) emergency overflow. These are competing objectives, as the first one tends to fill the pond while the latter one tends to empty it.

3. Preliminaries

We apply the mathematical modeling framework of hybrid Markov decision process (HMDP), adapted from [29,36], for modeling our problem.

Definition 1. A *hybrid Markov decision process* (HMDP) $\mathcal{M}$ is a tuple (C, U, X, F, δ) where:

- the controller C is a finite set of (controllable) modes $C = \{c_1, \dots, c_k\}$,
- the uncontrollable environment U is a finite set of (uncontrollable) modes $U = \{u_1, \dots, u_l\}$,
- $X = \{x_1, \dots, x_n\}$ is a finite set of continuous (real-valued) variables,
- for each $c \in C$ and $u \in U$, the flow function $F_{c,u} : \mathbb{R}_{>0} \times \mathbb{R}^X \to \mathbb{R}^X$ describes the evolution of the continuous variables over time in the combined mode (c, u), and
- δ is a family of density functions $\delta_\gamma : \mathbb{R}_{\geq 0} \times U \to [0, 1]$, where $\gamma = (c, u, \boldsymbol{x})$ is a global configuration with $\boldsymbol{x} : X \to \mathbb{R}$ being a valuation. More precisely, $\delta_\gamma(\tau, u')$ is the probability that in the global configuration $(c, u, \boldsymbol{x})$ the uncontrollable mode u will change to mode u' after a delay τ. Note that $\Sigma_{u'} \int_\tau \delta_\gamma(\tau, u')\mathrm{d}\tau = 1$.

This notion of an HMDP describes an uncountable and infinite state Markov Decision Process, see [37], where the controller mode switches periodically with interval $P \in \mathbb{R}_{\geq 0}$ and the uncontrollable environment mode switches probabilistically according to δ. In the rest of the paper, we denote by $\mathbb{C}$ the set of global configurations $C \times U \times (X \to \mathbb{R})$ of an HMDP.

The evolution of an HMDP over time is defined as follows. Let $\gamma = (c, u, \boldsymbol{x})$ be the current configuration, $\gamma' = (c', u', \boldsymbol{x}')$ the next configuration, and τ a time delay. We write $\gamma \xrightarrow{\tau} \gamma'$ when $c' = c$, $u' = u$, and $\boldsymbol{x}' = F_{c,u}(\tau, \boldsymbol{x})$. We write $\gamma \xrightarrow{\tau}_u \gamma'$ in case $c' = c$, $\boldsymbol{x}' = F_{c,u}(\tau, \boldsymbol{x})$ and $\delta_\gamma(\tau, u') > 0$.

A *run* of an HMDP is an interleaved sequence $\pi \in \mathbb{C} \times (\mathbb{R}_{\geq 0} \times \mathbb{C})^*$ of configurations and time-delays, starting with initial configuration γ_0:

$$\pi = \gamma_0 \tau_1 \gamma_1 \tau_2 \gamma_2 \tau_3 \cdots$$

where $\gamma_i = (c_i, u_i, \boldsymbol{x}_i)$, for all n there exist i such that $\Sigma_{j \leq i} \tau_j = n \cdot P$, and for all i either

1. the environment changes to a new mode, i.e., $\gamma_i \xrightarrow{\tau_{i+1}}_u \gamma_{i+1}$, or
2. the controller changes to any possible new mode when it reaches the end of a period, i.e., $\Sigma_{j \leq i+1} \tau_j$ is a multiple of P and $\gamma_i \xrightarrow{\tau_{i+1}} (c_{i+1}, u_{i+1}, \boldsymbol{x}_{i+1})$ with $c_{i+1} \in C$ and $u_{i+1} = u_i$.

The control problem of a stormwater detention pond can be described as an HMDP as follows. The set of control modes C contains the different pond outlet water brake settings that can be chosen. For static control, C becomes a singleton. The rain determines the uncontrollable stochastic input to the system, which is modeled with two uncontrollable modes: dry and raining. The density function δ captures the uncertainty in the duration of the dry and rain intervals, which is independent of c and $\boldsymbol{x}$. Finally, X contains the state variables, such as the current water level in the pond and the current rain intensity.

For the model of the detention pond, the flow function $F_{c,u}$ is expressed as a combination of differential equations and timed automata. A timed automaton [38] is a tuple $A = (Loc, l_0, Clk, E, Act, Inv)$, where Loc is a finite set of locations, $l_0 \in Loc$ is the initial location, Clk is a finite set of clocks, $E \subseteq Loc \times Act \times \mathcal{B}(Clk) \times 2^{Clk} \times Loc$ is a set of edges, Act is a finite set of actions, and $Inv : Inv \mapsto \mathcal{B}(Clk)$. Here $\mathcal{B}(Clk)$ is the set of all predicates using the clocks, and for an edge $e = (l_s, a, g, c, l_t)$ locations l_s and l_t represent the source location and target location, respectively, a is the action, g the guard, and c a set of clocks to be reset. When such a predicate is used on an edge, it is called a guard. Finally, 2^{Clk} indicates on edges the set of clocks that are reset to 0.

Example 1. Fig. 3 shows a small HMDP example to illustrate the used modeling formalism; the shown values are hypothetical and not based on real-life data. Rain can fall into a storage tank, from which water can be drained with a controlled valve. The uncontrollable environment is modeled with the (stochastic) timed automaton *Rain*. The two locations, depicted with circles, represent the two modes ‘`Dry`’ and ‘`Raining`’, and edges, depicted by arrows between locations, represent mode switches. The clock variable x keeps track of the duration of modes. The model indicates that the dry mode has a duration between 6 and 12 time units, as the invariant $x \leq 12$ indicates that the system can only remain in ‘`Dry`’ for at most 12 time units and the guard $x \geq 6$ on the outgoing edge indicates that the transition from ‘`Dry`’ to ‘`Raining`’ is only possible after 6 time units. The raining mode has a duration of b time units, which is sampled from an exponential distribution with mean i, a variable that increases with the number of rain cycles. Once the uncontrollable edge is taken from ‘`Dry`’ (the source location of that edge) to ‘`Raining`’ (the target location of that edge), the rain intensity r is chosen from a bounded normal distribution with mean 8, standard deviation 1, upper bound 10, and lower bound 6, all volume unit per time unit. In this paper, uncontrollable edges are indicated by dashed arrows. The initial

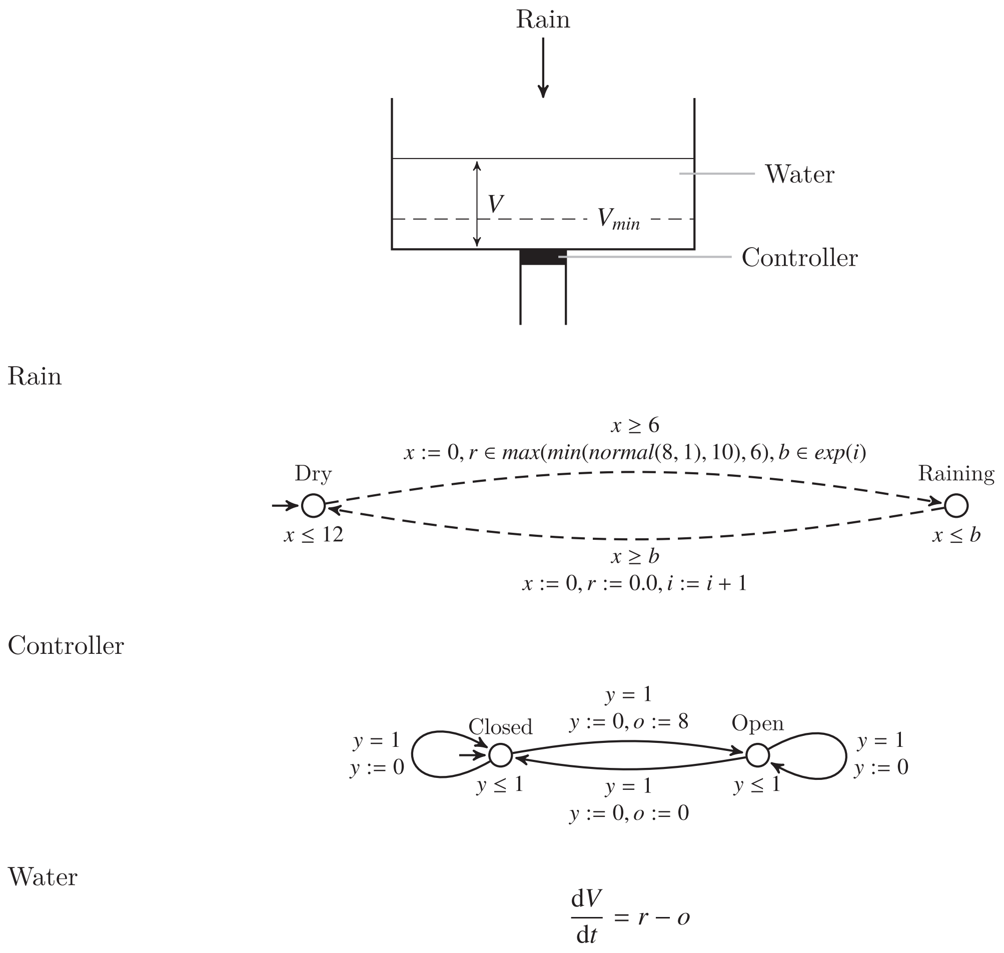


Fig. 3. A small HMDP example.

location is indicated by the small incoming arrow at location ‘Dry’. The initial value of clock variables is assumed to be 0 when not depicted.

The timed automaton *Controller* models the controllable valve, which is either in control mode ‘Closed’ or ‘Open’. The solid edges indicate controllable actions. Clock variable y keeps track of the control period duration, where the control period is set to 1 (see guards $y = 1$ on the edges). After each control period, the controller has the choice of switching to ‘Closed’ or to ‘Open’. When switching to the ‘Closed’ mode, the output flow o is set to 0 volume units per time unit, while switching to the ‘Open’ mode it is set to 8 volume units per time units.

Finally, the *Water* model describes the evolution of the water volume V over time with a differential equation: the volume change is the difference between the water inflow r and the water outflow o. For this example, the safety objective is to maintain a minimal water level V_{min}, while the optimization objective is to minimize the expected average (accumulated) water level.

3.1. Strategies for HMDP

For a given HMDP, a memoryless and possibly nondeterministic *strategy* σ determines which of the control modes can be used in the next period. Formally, a strategy is a function $\sigma : \mathbb{C} \to 2^C$ that returns a nonempty set of allowed control modes in a configuration. A strategy is called *deterministic* if exactly one control mode is permitted in each configuration.

The behavior of an HMDP $\mathcal{M}$ under supervision of a strategy σ, denoted as the stochastic process $\mathcal{M} \upharpoonright \sigma$, is defined as follows. A run π is *according to the strategy* σ if the controller changes mode according to the strategy σ, i.e., $\gamma_i \xrightarrow{\tau_{i+1}} (c_{i+1}, u_{i+1}, \boldsymbol{x}_{i+1})$ with $c_{i+1} \in \sigma((c_i, u_i, \boldsymbol{x}_{i+1}))$ and $u_{i+1} = u_i$. A strategy σ is called *safe* with respect to a set of configurations $S \subseteq \mathbb{C}$, called the safe set, if for any run π according to σ all configurations encountered are within the safe set S. Note that we require $\gamma_i \in S$ for all i and also

$\gamma_i' \in S$ whenever $\gamma_i \xrightarrow{\tau} \gamma_i'$ with $\tau \leq P$. A safe strategy is called *maximally permissive* if for each configuration it returns the largest set of possible actions [26].

The optimality of a strategy can be evaluated for the stochastic process $\mathcal{M} \upharpoonright \sigma$ with a given optimization variable. Let $H \in \mathbb{R}_{\geq 0}$ be a given time-horizon and D a random variable on finite runs, then $\mathbb{E}^{\mathcal{M},\gamma}_{\sigma,H}(D) \in \mathbb{R}_{\geq 0}$ is the expected value of D on the space of runs of $\mathcal{M} \upharpoonright \sigma$ of length H starting in configuration γ. For example, D can be the integrated error (or deviation) of a continuous variable with respect to its desired target value.

The goal is to synthesize a safe and optimal strategy σ_{opt} for a given HMDP $\mathcal{M}$, initial configuration γ, safety set S, optimization variable D, and time-horizon H. To obtain σ_{opt}, the tool Uppaal Stratego [26] can first synthesize a maximally permissive safe strategy σ_{safe} with respect to S when the HMDP $\mathcal{M}$ can be represented as a timed game with integer-valued bounds. Subsequently, σ_{opt} is a sub-strategy of σ_{safe} (i.e., $\forall \gamma \in \mathbb{C} : \sigma_{opt}(\gamma) \subseteq \sigma_{safe}(\gamma)$) that optimizes (minimizes or maximizes) $\mathbb{E}^{\mathcal{M},\gamma}_{\sigma,H}(D)$. For additive random variables, the optimal sub-strategy of the maximally permissive strategy is deterministic. Yet, when the system's dynamics is more complicated, e.g. described by differential equations, an alternative for synthesizing a safe strategy is to incorporate violations of the safety set S as severe penalties in the optimization variable D.

3.2. Uppaal Stratego

We use the modeling tool Uppaal Stratego [26] for control synthesis. It integrates Uppaal with the two branches Uppaal SMC [39] (statistical model checking for stochastic hybrid systems) and Uppaal Tiga [40] (synthesis for timed games). Therefore, Uppaal is able to synthesize safe *and* optimal strategies. To synthesize a safe and optimal strategy σ_{opt}, Uppaal Stratego first needs the HMDP $\mathcal{M}$ being abstracted (if possible) into a 2-player timed game, ignoring all stochasticity. A safe strategy σ_{safe} can then be synthesized for this timed game. A simplified version of timed computational tree logic (TCTL) [40] is used to formulate the safety specification. Subsequently, reinforcement learning is applied to obtain an optimal sub-strategy σ_{opt} based on $\mathcal{M} \upharpoonright \sigma_{safe}$ and the given random optimization variable [26]. When the HMDP $\mathcal{M}$ is not abstracted into a 2-player timed game, reinforcement learning can still be used to learn an optimal strategy that most likely is also safe, although no guarantees can be given. For this optimal strategy, statistical model checking can be applied to estimate the probability that the system under this strategy $\mathcal{M} \upharpoonright \sigma_{opt}$ leaves the set of safe states S.

Several learning algorithms are at the modelers disposal in Uppaal Stratego. Recently, in [31] Q-learning and M-learning were introduced. With Q-learning, sample runs are drawn from the HMDP model and are used afterwards to calculate the so-called Q-values. These values are refined into a new strategy and the previous step is repeated with this new strategy until some termination criteria are met. M-learning works similar to Q-learning, except that the HMDP model is now utilized to approximate the transition and cost functions, which are used to calculate the Q-values instead of sample runs. To efficiently cope with continuous state spaces, Uppaal Stratego deploys online partition refinement techniques.

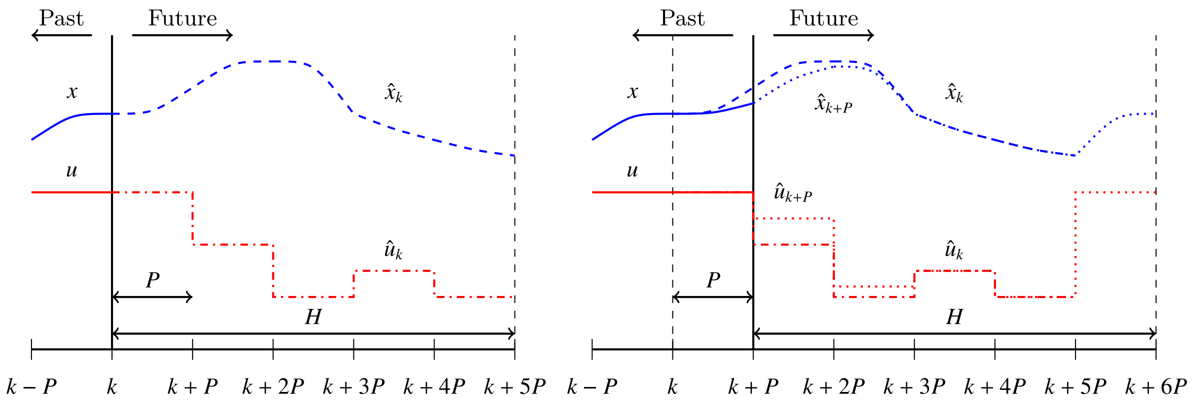
3.3. *Off-line and on-line control*

A strategy can be synthesized both off-line and on-line. With off-line control, a safe and optimal strategy is synthesized before it is fully implemented on the system. The time-horizon H has to be sufficiently large to capture all possible behaviors of the system that the strategy should handle. To illustrate the potential problems with small horizons H, consider again Example 1 in which the safety objective is to keep the water level always above the minimum level while the optimization objective is to minimize the average water level. One safe strategy is to keep the valve always closed such that the water level never drops below the minimum water level. If we keep $H < 6$ for off-line control, no run with rainfall is generated, thus reinforcement learning concludes that this safe strategy is also the optimal. Yet, when we deploy this strategy and let time run longer than this horizon, the strategy is no longer optimal as rainfall will eventually increase the water level.

In on-line control, also called model-predictive control (MPC) [32], the control strategy is recalculated periodically, typically after a single control period P, where the model is updated with the latest sensor readings. Fig. 4 illustrates the concept of MPC used in this paper. Up to time $t = k$, the true state of the system x has been observed and control input u has been provided to it. Using a model of the system, the future state $\hat{x}_k$ trajectory within a control horizon H is predicted. The evolution of the state depends on the control sequence being applied $\hat{u}_k$, where the applied control action can be switched after each control period. This control sequence can be the result of the safe and optimal strategy σ_{opt} from Section 3.1. Once the optimal control sequence is obtained, the first control action of this sequence is applied. When the end of the control period is reached, the process mentioned above is repeated. At time $t = k + P$ the true value of the state of the system $x(k + P)$ is observed, which is, most likely, different from the predicted state $\hat{x}_k(k + P)$, see the solid blue line in Fig. 4(b). Therefore, using the new true state $x(k + P)$ might result in a different control sequence $\hat{u}_{k+P}$ than the one calculated before $\hat{u}_k$.

4. Modeling

Our model of a detention pond consists of the components Pond, Controller, Rain, and UrbanCatchment. The model can be downloaded from [41].



(a) $t = k$ (b) $t = k + P$

Fig. 4. Conceptual overview of model predictive control. (a) shows the system at time $t = k$. In solid blue and dashed blue is the continuous evolution of the state in the past x and for the future $\hat{x}_k$, respectively, while solid red and dashdotted red shows the periodically switched control signal in the past u and for the future $\hat{u}_k$, respectively. Both future trajectories are predicted at time $t = k$ using some model. (b) shows the system after a period P at time $t = k + P$. New predictions $\hat{x}_{k+P}$ and $\hat{u}_{k+P}$ are made (dotted lines) based on the true state x at time $t = k + P$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

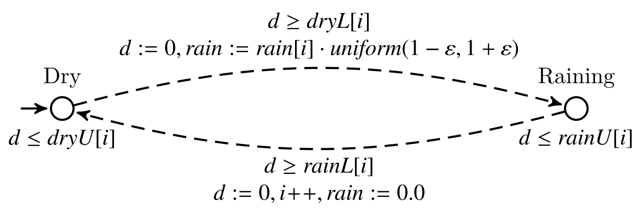

$d \geq dryL[i]$
$d := 0, rain := rain[i] \cdot uniform(1 - \varepsilon, 1 + \varepsilon)$
Dry
Raining
$d \leq dryU[i]$
$d \leq rainU[i]$
$d \geq rainL[i]$
$d := 0, i{+}{+}, rain := 0.0$

Fig. 5. The model generating rainfall.

4.1. Rain

Fig. 5 shows the rain model including its uncertainty. It generates the uncontrollable input to the system and can be interpreted as the weather forecast. The rain profile is modeled as alternating dry and raining intervals, each modeled with a location. For each interval period, indicated with i, the duration of the dry (raining) period is bounded between $dryL[i]$ ($rainL[i]$) and $dryU[i]$ ($rainU[i]$), all being positive integers. Clock d tracks the duration of the current interval. The actual dry or rain duration is chosen uniformly at random between $dryL[i]$ and $dryU[i]$ or $rainL[i]$ and $rainU[i]$, respectively.

When it is raining in the ith interval, the actual rain intensity $rain \in \mathbb{R}$ used as input for the UrbanCatchment model is chosen uniformly random between $rain[i] \cdot (1 - \varepsilon)$ and $rain[i] \cdot (1 + \varepsilon)$, where ε is a fixed uncertainty factor. Within an interval, the rain intensity is constant.

The concrete values for $dryL[i], dryU[i], rainL[i], rainU[i]$, and $rain[i]$ are derived from historical rain data from the [42], as historical weather forecasts are not published.

4.2. Urban catchment

We model the urban catchment area as a one-layer linear reservoir model (the surface storage of the simplified 'Nedbør Afstrømnings Model' (NAM)), see [43]. It is assumed that both the $rain$ and the stored rain water $S \in \mathbb{R}$ are uniformly distributed along the urban area, so both become a height measure instead of volume. The time-dependent dynamics of S is given by

$$\frac{\mathrm{d}S}{\mathrm{d}t} = rain - kS, \tag{1}$$

where k is the urban surface reaction factor. This expression simply states that the change in stored water S depends on the difference between the rain falling into the urban area and the storm water leaving it. The flow (expressed as a volume per time unit) from the urban catchment to the pond Q_{in} is given with the rational method by

$$Q_{in} = kSA_{uc} \tag{2}$$

with A_{uc} being the urban catchment surface area.

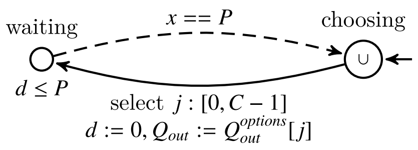


Fig. 6. The model of the controller. The **select** statement is a simplification of having a set of edges with one for each value in the select range.

4.3. Pond

The dynamics of the water level w in the pond can be derived using the mass balance equation. Assuming constant density of water, this translates into a volume balance equation. Using Fig. 2, we see that the difference in inflow and outflow contribute to the change in water inside the pond. Therefore,

$$Q_{in} - Q_{out} = \frac{\mathrm{d}V}{\mathrm{d}t}, \tag{3}$$

where Q_{in} is the water inlet flow from the urban drainage system, Q_{out} is the water outlet flow into a nearby stream, and V the water volume of the pond above the permanent water level. The outlet flow is assumed to be more or less constant for any chosen output mode (see Section 4.4), but in reality there is a non-linear relationship to the water level, especially when the outlet is not fully submerged with low water levels. This is however simplified in this model to reduce computational effort for Q-learning.

The change in volume over time can also be expressed using the geometry of the pond:

$$\frac{\mathrm{d}V}{\mathrm{d}t} = \frac{\mathrm{d}(wA_p(w))}{\mathrm{d}t}, \tag{4}$$

where $A_p(w)$ is the pond surface area at height w. Eqs. (3) and (4) together describe the dynamics of the pond's water level under 'normal' circumstances.

There are two boundary cases that need to be taken into account. The first case is when the outflow is larger than the inflow and the water level reaches the permanent water level. The second case is when the inflow is larger than the outflow and the water level reaches the maximum height W of the pond, which results in an emergency overflow. In both cases, the water level w remains stationary. Now, Eq. (3) can be reformulated taking these boundary cases into account:

$$\frac{\mathrm{d}V}{\mathrm{d}t} = \begin{cases} 0 & \text{if } Q_{out} \geq Q_{in} \wedge w = 0, \\ 0 & \text{if } Q_{in} \geq Q_{out} \wedge w = W, \\ Q_{in} - Q_{out} & \text{otherwise.} \end{cases} \tag{5}$$

Note that if one would time-discretize the above differential equations, the models from [20,22] can be obtained.

4.4. Controller

The controller is able to change the size of the pond's outlet periodically. Fig. 6 shows the model of the controller. It starts in the urgent location '`choosing`' from where it chooses with a controllable action one of the C control options. Location '`choosing`' is urgent to force the periodic controller to make a choice without letting time pass in this location, so the controller is truly periodically. The actual output Q_{out} is set to one of the predefined constant outputs for each mode $Q_{out}^{options}$. The clock d, measuring the duration of the current control period, is reset to 0. Note that d is local to the controller and does not interfere with clock d from the Rain model.

In location '`waiting`' the controller waits until the period with duration P is over, indicated with invariant $d \leq P$. When the controller has waited for P time units, it goes to the right urgent location and above process is repeated for the next control period.

Note that in our model we require a discrete number of control options (a discrete action space) such that we can use reinforcement learning as implemented in Uppaal Stratego. On the contrary, the works of [20,22] allow a continuous action space. But since we allow the controller to switch periodically, could create new average outflows. For example, if the available outflows to choose from are 1 L/s and 2 L/s while the optimal outflow would be 1.5 L/s, switching evenly between 1 L/s and 2 L/s creates an average outflow over time of 1.5 L/s.

4.5. Control problem definition

The primary objective of the controller is to ensure a safe operation of the stormwater detention pond. In this context, safety is defined as preventing the pond from overflowing. In case of an overflow event, the water discharge in the nearby stream or river is temporarily much higher than normal. This excessive discharge might have environmental impacts or cause downstream flooding. We measure overflow with a continuous variable o that represents the accumulated overflow duration. Formally,

$$\frac{\mathrm{d}o}{\mathrm{d}t} = \begin{cases} 1 & \text{if } w = W, \\ 0 & \text{if } w < W. \end{cases} \tag{6}$$

Table 1
The rain forecast data for September 5–September 7 2019.

I	$dryL$ [min]	$dryU$ [min]	$rainL$ [min]	$rainU$ [min]	$rain$ [cm/min]
1	210	256	27	33	0.01333
2	64	78	21	25	0.03478
3	1376	1682	49	61	0.02545
4	168	206	23	29	0.02308
5	203	249	208	254	0.00952

The secondary objective is to capture as much urban area particles as possible from the storm water. This is done to prevent contamination of the nearby stream or river. Particles are captured through particle sedimentation onto the pond's floor surface. Therefore, we want to maximize the retention time t_r, defined as $t_r = \frac{V}{Q_{out}}$. To maximize the retention time, we can maximize V or minimize Q_{out}, or both. Since V is proportional to w, maximizing V implies maximizing w. From Eq. (3) it follows that minimizing Q_{out} results in a larger $\frac{\mathrm{d}V}{\mathrm{d}t}$, i.e., over time a larger V is encouraged. Therefore, penalizing a low water level encourages a larger water volume V and a lower outflow Q_{out}.

In the model, we associate a cost c to the ability of particle sedimentation. A linear cost function is used such that higher water levels, related to higher possibilities for particle sedimentation, result in lower cost. Formally,

$$\frac{\mathrm{d}c}{\mathrm{d}t} = 1 - \frac{w}{W}. \tag{7}$$

Therefore, c represents the accumulated cost, where a cost of 1 per time unit is associated with w being the permanent water level and a cost of 0 with w being at the maximum height W.

Combining both objectives, the controller synthesis problem can now be formulated as follows. Synthesize a safe and optimal strategy σ_{opt} such that it minimizes overflow duration and particle sedimentation cost. The optimal strategy formulation becomes

$$\sigma_{opt} = \underset{\sigma}{\operatorname{argmin}} \ \mathbb{E}^{\mathcal{M},\gamma}_{\sigma,H}(c + \alpha o), \tag{8}$$

where $\alpha \in \mathbb{R}$ is the overflow penalty factor to balance the contribution of both o and c to the optimization. Uppaal Stratego is applied to synthesize controllers for this problem, where the continuous variables o and c are implemented in a separate component and Eq. (8) is the optimization query.

5. Off-line controller synthesis

For the off-line control simulation experiments, we calibrated our model to the Vilhelmsborg Skov pond south of Aarhus, Denmark. It has an urban catchment area of $A_{uc} = 0.59$ ha, a maximally permitted discharge of 95 L/s, and an average pond area $A_p = 5,572$ m^2 (data from [35]). We estimated the urban surface reaction factor to $k = 0.25$ and the maximum water level to $W = 300$ cm.

Historical rain data for the period September 5–September 7 2019 are used, obtained from [42]. We artificially created a weather forecast based on the actual historical rainfall. We performed the following steps to create a simple weather forecast including prediction uncertainties. For each rain period (a continuous period of reported non-zero rain intensity), we average the rain intensity[2]. Subsequently, an uncertainty of $\epsilon = 10\%$ is added to the observed interval durations and rain intensities to mimic a weather forecast. In this period, five rain periods occurred with varying rainfall durations and intensities.

Table 1 shows the obtained rain data implemented in the model, where September 5 starts dry (so the first rain starts falling between 3.30 am and 4.16 am and has a duration between 27 and 33 min).

Fig. 7 shows the results of ten simulated runs in Uppaal Stratego with an initial water height $w = 100$ cm and the current static control strategy, i.e., the number of control modes C for the water brake is 1. We set the static output flow to 2/3 of the permitted output flow. In blue and solid lines the water level is plotted, in black and dotted lines the rain. The discretization step for the simulations is set to 0.5 min. We observe that several of the ten runs eventually result in an emergency overflow of the pond at the time between 2600 and 2800 min (7-11 pm on September 6). This is also confirmed by analyzing the expected value of o: $\mathbb{E}^{\mathcal{M},w=100}_{\sigma_{static},3\text{ days}}(o) = 2.1 \pm 0.5$ (95% CI), i.e., the pond is expected to be overflowing for 2.1 min.

An actively controlled water brake can have three different modes: small, medium, and large. We set the medium setting to the current static output flow capacity of $2/3 \cdot 95$ L/s. The low setting is 0.25 times medium and high 1.5 times medium, thus the high setting corresponds to the maximally permitted discharge of 95 L/s. Due to energy constraints (these stormwater ponds are not connected to the normal electric energy grid), the water brake can only change once every hour, so $P = 60$ min. We employ Q-learning to synthesize strategies with the learning parameters set to 50 successful runs, a maximum of 100 runs, 20 good runs, and 20 runs to evaluate (the first four learning parameters in Uppaal Stratego).

With these control modes, we synthesized an optimal controller using Eq. (8) with the overflow penalty factor $\alpha = 10000$. Fig. 8 shows ten runs in Uppaal Stratego of the model using the synthesized optimal controller. As can be seen from the figure, in order

[2] By averaging the rain intensities, we also resolve the fixed first data-point of each rain period due to the tipping bucket rain gauge design.

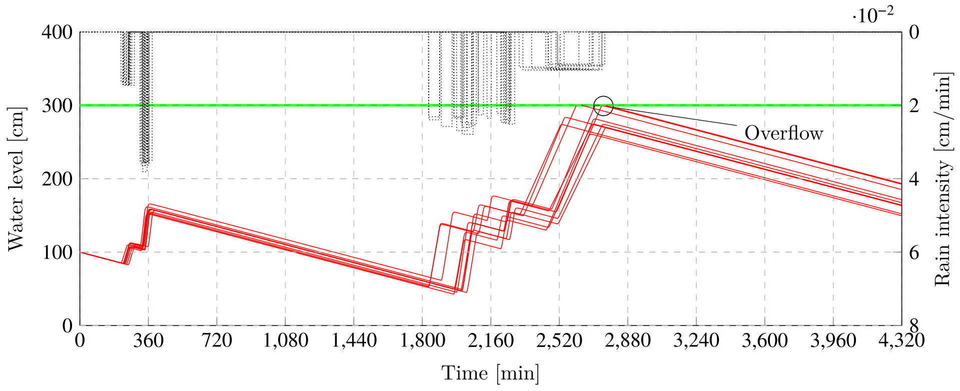


Fig. 7. Ten simulations of the model with the current static control. Blue and solid lines indicate the water level in the pond and black and dotted lines the rainfall. The horizontal red line indicates the maximum water level. For one of the runs, overflow occurs around $t = 2750$ min, as the maximum water level is $W = 300$ cm. As a quick reference, 1440 min is 1 day, and the total time scale is 3 days. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

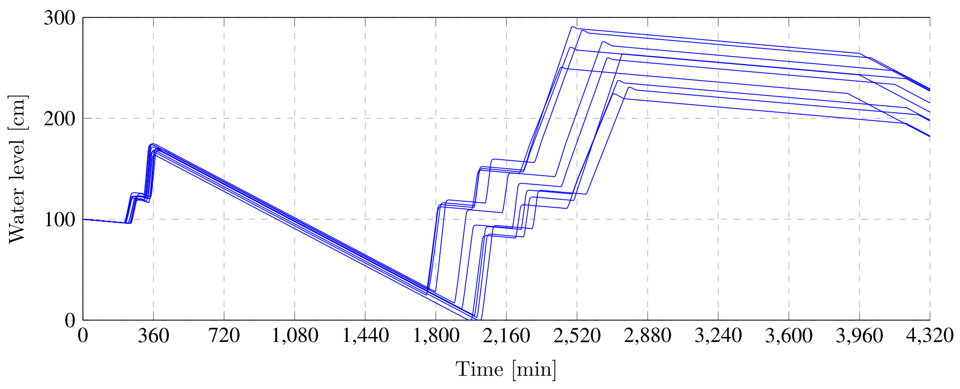


Fig. 8. Ten simulations of the model with optimal dynamic control. As a quick reference, 1440 min is 1 day, and the total time scale is 3 days.

to ensure safety, the controller keeps the water level in the pond lower than the static controller (see Fig. 7 for the static controller results). For some runs the pond water level even reaches the permanent water level, i.e., it cannot go lower. While overflow is not guaranteed to be avoided, including it in the controller's cost function, see Eq. (8), reduces the probability of overflow significantly to between 0% and 5% (95% CI). For static control the probability of overflow is between 10% and 19% (95% CI). Furthermore, the expected overflow duration with the synthesized strategy is calculated as $\mathbb{E}^{\mathcal{M},w=100}_{\sigma_{opt},3\text{ days}}(o) \approx 0$ (95% CI), also indicating that overflow is avoided with the synthesized strategy. Yet, having a safe strategy comes with higher cost for particle sedimentation. For static control, the expected cost is $\mathbb{E}^{\mathcal{M},w=100}_{\sigma_{static},3\text{ days}}(c) = 2015 \pm 8$ (95% CI), while for the optimal control it is $\mathbb{E}^{\mathcal{M},w=100}_{\sigma_{opt},3\text{ days}}(c) = 2157 \pm 11$ (95% CI). This is an increase of 7%, but accepted as our primary aim is to avoid flooding.

Fig. 9 shows simulation results for the case that the initial water level is set to 0 cm. We notice that the synthesized optimal strategy now lowers the output water brake setting compared to the previous experiment, as the pond has sufficient capacity for the upcoming rain. As no guaranteed safe strategy is synthesized, the probability of overflow is between 0% and 7.7% (95% CI). For static control the probability of overflow is between 0% and 5% (95% CI). So there is a slight increase in the probability of overflow. This can also be observed from the figure, as the optimal strategy tries to maintain a higher water level for particle sedimentation. The cost for particle sedimentation also indicates this. Optimal control results in a particle sedimentation cost improvement of 26% compared to static control: $\mathbb{E}^{\mathcal{M},w=0}_{\sigma_{static},3\text{ days}}(c) = 2945 \pm 6$ (95% CI) for static control and $\mathbb{E}^{\mathcal{M},w=0}_{\sigma_{opt},3\text{ days}}(c) = 2189 \pm 8$ (95% CI) for optimal dynamic control.

Finally, using Uppaal SMC we can analyze statistically whether the controller might prevent overflow. For example, for the initial water level of $w = 100$ cm, the probability of overflow after three days with a random controller that can switch between the three settings is between 30% and 40%. So this indicates that there are strategies that can prevent overflow. If we increase the initial water level to, for example, $w = 150$ cm, this probability of overflow is between 95% and 100% (95% CI), i.e., no matter what the strategy is, it is very likely that the pond overflows.

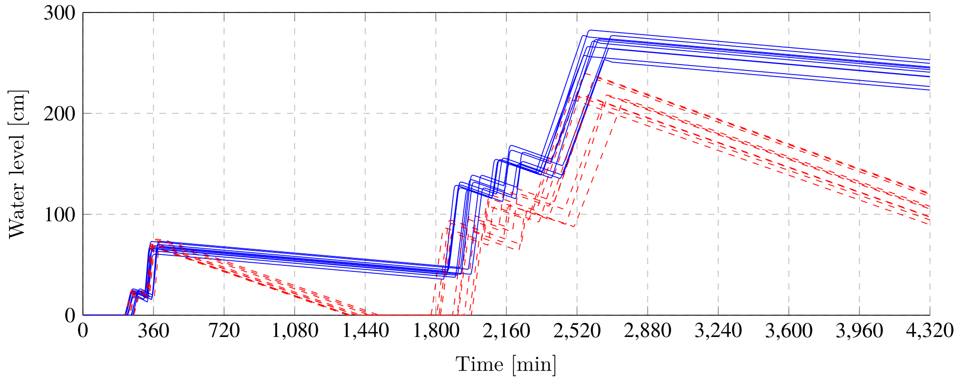


Fig. 9. Results for initial water level $w = 0$ cm. In blue and dashed lines are ten simulations of the model with static control and in red and solid lines ten with optimal dynamic control. As a quick reference, 1440 min is 1 day, and the total time scale is 3 days. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

6. On-line controller synthesis

6.1. Experimental setup

For on-line control, we use the U.S. Environmental Protection Agency Stormwater Management Model (EPA-SWMM) [34] to simulate the real world to provide updated sensor readings for the HMDP learning model. EPA-SWMM is an open-source physics-based dynamic rainfall-runoff model that has been implemented for decades in the urban stormwater management with respect to both hydrological and hydraulic simulations. EPA-SWMM can precisely trace the dynamics of flow conditions (e.g. stream flow and stormwater pond water level) in any given position of the real physical systems. Running a full SWMM model simulation can be computational expensive, which can be problematic in the context of reinforcement learning, which needs more than thousands of runs. Therefore, reinforcement learning utilizes the simplified HMDP model. We use pySWMM [44] for the interfacing of EPA-SWMM with Python. Finally, we combine pySWMM with Uppaal Stratego using the STOMPC framework [45][3], which facilitates the setup of model-predictive control using Uppaal Stratego and an external simulator. To summarize, the simple HMDP model, as implemented in Uppaal Stratego, is used to synthesize controllers, while the EPA-SWMM model is used to analyze the performance of the synthesized controller as if it was applied in a real pond.

For online control, we calibrated the models to a pond with a surface area of $A_p = 2,250$ m^2, a maximum water level of $W = 200$ cm, and a maximally permitted discharge of 100 L/s. Its urban catchment has an area of $A_{uc} = 20$ ha and a surface reaction factor of $k = 0.25$ (only used in the HMDP model in Uppaal Stratego for learning; EPA-SWMM deploys a more sophisticated model to simulate the flow from an urban catchment area).

In EPA-SWMM, an orifice is used as a link that can dynamically control the outflow of the pond by changing the diameter of the opening of the orifice. We selected a circular opening of the orifice. Following the documentation of EPA-SWMM [46], the outflow of the pond becomes

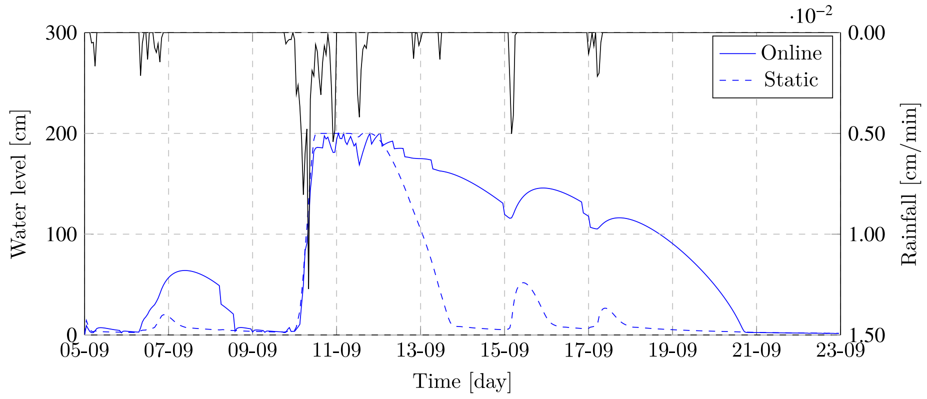
$$Q_{out} = C\frac{\pi}{4}d^2\sqrt{2gw} \tag{9}$$

with C the discharge coefficient, d the chosen diameter of the orifice, and g the gravitational acceleration. Note that this relationship is only valid if the orifice is fully submerged, i.e. $w \geq d$ (see Fig. 2 for a reference). In our Uppaal Stratego model we implement this relationship ignoring the partially submerged state to keep it simple. This simplification accelerates the run generation with acceptable accuracy for Q-learning.

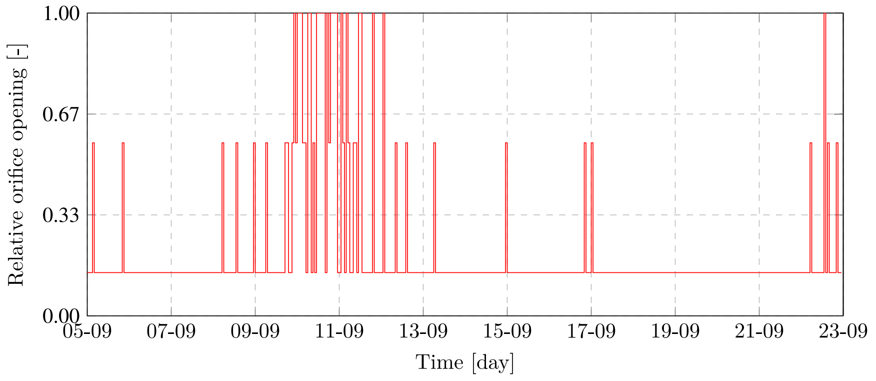
The controller can again choose from three different settings: low, medium, and high. The medium setting is considered to be the static control setting. As we now have to provide EPA-SWMM with a diameter setting of the orifice, we set the high diameter setting to $d_h = 17,5$ cm, medium to $d_m = 4/7 \cdot d_h$, and low to $1/7 \cdot d_h$. This results in maximum outflows of 98 L/s (which is just below the maximum permitted discharge), 32 L/s, and 2 L/s, respectively. We set the control period to $P = 1$ h and the control horizon to $H = 12$ h. This differs from [20], in which a control period of 5 min and a control horizon of 30 min are used, and [22], in which a control period of 1 day and a control horizon of 7 days are used.

Weather forecasts are again derived from historical rain data from [42]. We apply the same method as in Section 5 for the construction of the weather forecasts, but now we perform this construction at the beginning of each control period to update it. We let our on-line simulations run from September 5, 2019 to September 23, 2019, spanning a period of 2.5 weeks. This period includes some average rainfall, some heavy rainfall, and a long dry period. All other settings are the same as in Section 5.

[3] See https://strategoutil.readthedocs.io/en/latest/



(a)



(b)

Fig. 10. Simulation results of online control. The top plot shows the water level in the pond for on-line model-predictive control (blue, solid) compared to static control (blue, dashed), where 200 cm is maximum water level. The black line shows the actual hour-average rainfall intensity, plotted with a reversed y-axis. The bottom plot shows the chosen relative orifice diameter setting, where 1 corresponds to d_h. The control setting is only allowed to be changed once every hour.

6.2. Simulation results

Fig. 10 shows the simulation results of using on-line control for the stormwater pond compared to a static control strategy. The water levels and rainfall are the outputs of and the inputs to the detailed EPA-SWMM model and not the simplified Uppaal Stratego model. For the periods with light or no rainfall (September 5–September 10 and September 12–23), the synthesized optimal controller keeps much more water in the pond compared to static control, which benefits particle sedimentation. The controller can do this, as the predicted rainfall over the next control horizon does not fill up the pond completely. This results in a total particle sedimentation cost reduction of 29% compared to the static control strategy.

During the heavy rainfall between September 10–September 12, both controllers have difficulties preventing the pond from overflowing, albeit that the synthesized on-line controller reduces the overflow duration significantly. The static controller results in an overflow duration of 24.1 h, while the optimal on-line controller results in an overflow duration of 1.2 h. This can also be seen in Fig. 10, where the blue solid line often is just below the maximum water level of 200 cm, and only overflows briefly at the starts of September 11 and September 12. Similar positive improvements of using online control, albeit for different scenarios, are observed in [20–23].

Fig. 10 also shows room for future improvement. There are several moments when the controller switches from low to medium and then back to low, see e.g. during the heavy rainfall and at September 15. At these instances, the pond has sufficient capacity for the upcoming rain, yet the controller decides to increase the outflow. The suspected causes for this are the control horizon and the learning budget assigned to Uppaal Stratego to be used for Q-learning. With a low control horizon, the synthesized controller might not be able to observe changes in the weather on time. With a low budget, the number of generated runs is low, thus not allowing Q-learning to observe all possible behavior to determine what is optimal. For this paper, we lowered the learning budget to simulate 2.5 weeks in reasonable time: each cycle of calculating an optimal controller with Uppaal Stratego and simulating the next hour with EPA-SWMM takes roughly 8.5 s. Another possible solution could be to include control mode switching explicitly in the cost function [22].

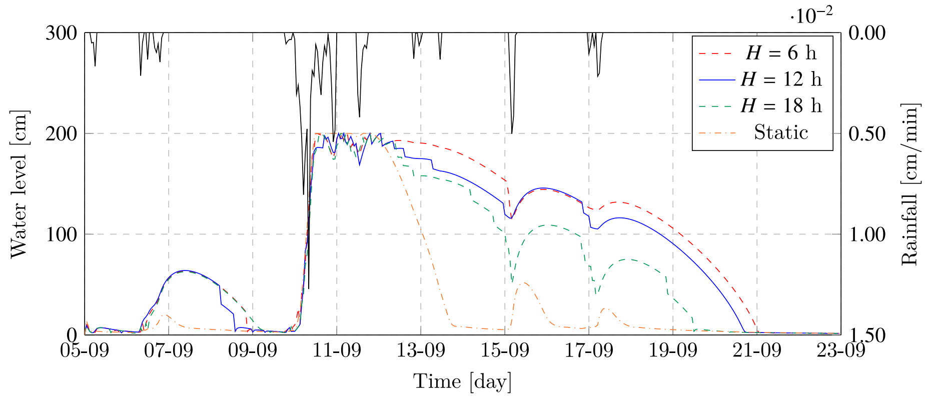


Fig. 11. Simulation results of online control with different control horizons while keeping the learning budget the same. Note that $H = 12$ h is the same result as in Fig. 10.

6.3. *Changing the prediction horizon*

Fig. 11 shows simulation results where three different control horizons H are used: 6 h, 12 h, and 18 h. All other parameters, including the learning budget, are kept the same. Intuitively, longer horizons are expected to achieve better results (as the learning algorithm can 'see' longer into the future); yet this relationship is more complex as a longer control horizon also implies more possible control strategies. After September 13, the results show that the pond's water level with $H = 6$ h is often higher than the one with $H = 12$ h, which in turn is higher than the water level with $H = 18$ h, as $H = 6$ h is "short sighted" to identify the coming rainfall events. Thus the synthesized controller with $H = 6$ h decided to reduce the valve opening accounting for "dry periods" in order to maximize the sedimentation. As a consequence, the controlled behavior of the pond with $H = 6$ h resulted in an immediate overflow at the start of the heavy rain on September 10, thus having a worse performance with respect to emergency overflow compared to the controllers obtained with longer control horizons.

The cause for these results also lies within the fixed budget for the generation of runs for Q-learning. For a certain control horizon H and control period P, the number of possible strategies is $3^{H/P}$, as each control period the controller can choose from three options. Ideally, for each possible strategy, at least one run should be generated by Uppaal Stratego so Q-learning can adjust the Q-table accordingly. When fewer runs are generated, it may happen that the optimal behavior is not 'observed' by Q-learning. As the number of possible strategies grows exponentially with the control horizon duration, having a long horizon can be detrimental for the computation time.

Managing and balancing the performance of the synthesized controllers and computational resource demand is a challenge for real-life applications. On the one hand there is sufficient computation time when the control period is set to 1 h or even 15 min: in the experiments the calculation of each controller took roughly 8.5 s, which is well within the deadline dictated by the control period. On the other hand, the computational power of a personal computer might not be a realistic representation of available computational resources in practice, as ponds can be located remotely and are often not connected to the regular power grid.

7. Conclusion and future challenges

We applied formal controller synthesis to automatically derive controllers for stormwater detention ponds where the water discharge into the nearby stream can be regulated. We showed that the problem can be modeled as a hybrid Markov decision process, such that symbolic and reinforced learning techniques from Uppaal Stratego can be applied. Simulation results of both off-line and on-line controls show that the synthesized near-optimal active controllers can outperform current static controllers. With on-line control the ever changing weather forecasts can be used to update synthesized strategies periodically.

This first step opens several future research directions. First, while the synthesized controller reduce the expected overflow duration significantly, no safety guarantee can be given. If the HMDP model of the pond can be transformed into a timed game with integer bounds, Uppaal Tiga can synthesize a guaranteed safe controller. Second, to increase the explainability of the synthesized strategies, it is to be investigated whether exporting strategies to decision trees, see [36], is possible. Third, it will be interesting to validate the approach with real-life data, especially using actual historical weather forecasts once they are published by DMI. We are currently in the process of doing this. Finally, only a single storm water detention pond is analyzed in isolation from the discharge stream. It would be interesting to see whether collaborative strategies can be synthesized for a collection of detention ponds all discharging into the same stream, as [21,23] show promising results for collaborative control.

CRediT authorship contribution statement

Martijn A. Goorden: Writing – original draft, Methodology, Investigation. **Kim G. Larsen:** Writing – review & editing, Supervision, Conceptualization. **Jesper E. Nielsen:** Writing – review & editing, Conceptualization. **Thomas D. Nielsen:** Writing – review & editing, Conceptualization. **Weizhu Qian:** Writing – review & editing. **Michael R. Rasmussen:** Writing – review & editing, Supervision, Conceptualization. **Jiří Srba:** Writing – review & editing, Conceptualization. **Guohan Zhao:** Writing – review & editing, Methodology.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The models and simulation data are available on Zenodo through https://doi.org/10.5281/zenodo.6592380.

References

[1] A. Semadeni-Davies, C. Hernebring, G. Svensson, L.-G. Gustafsson, The impacts of climate change and urbanisation on drainage in Helsingborg, Sweden: Suburban stormwater, J. Hydrol. 350 (1) (2008) 114–125, http://dx.doi.org/10.1016/j.jhydrol.2007.11.006.

[2] N. Alamdari, D.J. Sample, A.C. Ross, Z.M. Easton, Evaluating the impact of climate change on water quality and quantity in an urban watershed using an ensemble approach, Estuar. Coasts 43 (1) (2020) 56–72, http://dx.doi.org/10.1007/s12237-019-00649-4.

[3] T. Hvitved-Jacobsen, J. Vollertsen, A.H. Nielsen, Urban and Highway Stormwater Pollution: Concepts and Engineering, CRC Press, 2010, http://dx.doi.org/10.1201/9781439826867.

[4] G. Tixier, M. Lafont, L. Grapentine, Q. Rochfort, J. Marsalek, Ecological risk assessment of urban stormwater ponds: literature review and proposal of a new conceptual approach providing ecological quality goals and the associated bioassessment tools, J. Ecol. Indic. 11 (6) (2011) 1497–1506, http://dx.doi.org/10.1016/j.ecolind.2011.03.027.

[5] C.J. Walsh, A.H. Roy, J.W. Feminella, P.D. Cottingham, P.M. Groffman, R.P. Morgan, The urban stream syndrome: current knowledge and the search for a cure, JNABS 24 (3) (2005) 706–723, http://dx.doi.org/10.1899/04-028.1.

[6] C.J. Walsh, T.D. Fletcher, M.J. Burns, Urban stormwater runoff: A new class of environmental flow problem, PLoS One 7 (9) (2012) e45814, http://dx.doi.org/10.1371/journal.pone.0045814.

[7] DANVA, Administrationspraksis for regnvandsbassiner og udledningstilladelser, 2018 (in Danish).

[8] D.M.R. Jensen, A.T.H. Thomsen, T. Larsen, S. Egemose, P.S. Mikkelsen, From EU directives to local stormwater discharge permits: A study of regulatory uncertainty and practice gaps in Denmark, Sustainability 12 (16) (2020) 6317, http://dx.doi.org/10.3390/su12166317.

[9] J.T. Mobley, T.B. Culver, Design of outlet control structures for ecological detention ponds, J. Water Resour. Plan. Manag. 140 (2) (2014) 250–257, http://dx.doi.org/10.1061/(ASCE)WR.1943-5452.0000266.

[10] European Commission, Directive 2000/60/EC of the European Parliament and of the Council of 23 October 2000 Establishing a Framework for Community Action in the Field of Water Policy, 2000, Official Journal L 327.

[11] M. Schütze, A. Campisano, H. Colas, W. Schilling, P.A. Vanrolleghem, Real time control of urban wastewater systems–where do we stand today? J. Hydrol. 299 (3–4) (2004) 335–348, http://dx.doi.org/10.1016/j.jhydrol.2004.08.010.

[12] J.L. Webber, T. Fletcher, R. Farmani, D. Butler, P. Melville-Shreeve, Moving to a future of smart stormwater management: A review and framework for terminology, research, and future perspectives, Water Res. 218 (2022) 118409, http://dx.doi.org/10.1016/j.watres.2022.118409.

[13] B.R. Haverkort, M. Kuntz, A. Remke, S. Roolvink, M.I.A. Stoelinga, Evaluating repair strategies for a water-treatment facility using arcade, in: Dependable Systems and Networks, 2010, pp. 419–424, http://dx.doi.org/10.1109/DSN.2010.5544290.

[14] H. Hoppe, S. Messmann, A. Giga, H. Gruening, A real-time control strategy for separation of highly polluted storm water based on UV-Vis online measurements - from theory to operation, J. Water Sci. Technol. 63 (10) (2011) 2287–2293, http://dx.doi.org/10.2166/wst.2011.164.

[15] C. Sun, B. Joseph-Duran, T. Maruejouls, G. Cembrano, J. Meseguer, V. Puig, X. Litrico, Real-time control-oriented quality modelling in combined urban drainage networks, IFAC-PapersOnLine 50 (1) (2017) 3941–3946, http://dx.doi.org/10.1016/j.ifacol.2017.08.142.

[16] J.L. Svensen, C. Sun, G. Cembrano, V. Puig, Chance-constrained stochastic MPC of Astlingen urban drainage benchmark network, Control Eng. Pract. 115 (2021) 104900, http://dx.doi.org/10.1016/j.conengprac.2021.104900.

[17] D. Muschalla, B. Vallet, F. Anctil, P. Lessard, G. Pelletier, P.A. Vanrolleghem, Ecohydraulic-driven real-time control of stormwater basins, J. Hydrol. 511 (2014) 82–91, http://dx.doi.org/10.1016/j.jhydrol.2014.01.002.

[18] E. Gaborit, D. Muschalla, B. Vallet, P.A. Vanrolleghem, F. Anctil, Improving the performance of stormwater detention basins by real-time control using rainfall forecasts, Urban Water J. 10 (4) (2013) 230–246, http://dx.doi.org/10.1080/1573062X.2012.726229.

[19] E. Gaborit, F. Anctil, G. Pelletier, P.A. Vanrolleghem, Exploring forecast-based management strategies for stormwater detention ponds, Urban Water J. 13 (8) (2016) 841–851, http://dx.doi.org/10.1080/1573062X.2015.1057172.

[20] S. Shishegar, S. Duchesne, G. Pelletier, An integrated optimization and rule-based approach for predictive real time control of urban stormwater management systems, J. Hydrol. 577 (2019) 124000, http://dx.doi.org/10.1016/j.jhydrol.2019.124000.

[21] S. Shishegar, S. Duchesne, G. Pelletier, R. Ghorbani, A smart predictive framework for system-level stormwater management optimization, J. Environ. Manag. 278 (2021) 111505, http://dx.doi.org/10.1016/j.jenvman.2020.111505.

[22] W.D. Xu, T.D. Fletcher, M.J. Burns, F. Cherqui, Real time control of rainwater harvesting systems: The benefits of increasing rainfall forecast window, Water Resour. Res. 56 (9) (2020) e2020WR027856, http://dx.doi.org/10.1029/2020WR027856.

[23] W.D. Xu, M.J. Burns, F. Cherqui, K. Smith-Miles, T.D. Fletcher, Coordinated control can deliver synergies across multiple rainwater storages, Water Resour. Res. 58 (2) (2022) e2021WR030266, http://dx.doi.org/10.1029/2021WR030266.

[24] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197, http://dx.doi.org/10.1109/4235.996017.

[25] B.D. Bowes, C. Wang, M.B. Ercan, T.B. Culver, P.A. Beling, J.L. Goodall, Reinforcement learning-based real-time control of coastal urban stormwater systems to mitigate flooding and improve water quality, Environ. Sci.: Water Res. Technol. 8 (10) (2022) 2065–2086, http://dx.doi.org/10.1039/D1EW00582K.

[26] A. David, P.G. Jensen, K.G. Larsen, M. Mikučionis, J.H. Taankvist, Uppaal Stratego, in: Tools and Algorithms for the Construction and Analysis of Systems, in: Lecture Notes in Computer Science, 2015, pp. 206–211, http://dx.doi.org/10.1007/978-3-662-46681-0_16.

[27] E.R. Wognsen, B.R. Haverkort, M. Jongerden, R.R. Hansen, K.G. Larsen, A score function for optimizing the cycle-life of battery-powered embedded systems, in: Formal Modeling and Analysis of Timed Systems, in: Lecture Notes in Computer Science, 2015, pp. 305–320, http://dx.doi.org/10.1007/978-3-319-22975-1_20.
[28] K.G. Larsen, M. Mikučioni, J.H. Taankvist, Safe and optimal adaptive cruise control, in: Olderog-Festschrift, in: Lecture Notes in Computer Science, 2015, pp. 260–277, http://dx.doi.org/10.1007/978-3-319-23506-6_17.
[29] K.G. Larsen, M. Mikučioni, M. Muñiz, J. Srba, J.H. Taankvist, Online and compositional learning of controllers with application to floor heating, in: Tools and Algorithms for the Construction and Analysis of Systems, in: Lecture Notes in Computer Science, 2016, pp. 244–259, http://dx.doi.org/10.1007/978-3-662-49674-9_14.
[30] S. Bøgh, P. Gjøl Jensen, M. Kristjansen, K. Guldstrand Larsen, U. Nyman, Distributed fleet management in noisy environments via model-predictive control, in: Proceedings of the International Conference on Automated Planning and Scheduling, Vol. 32, No. 1, 2022, pp. 565–573, http://dx.doi.org/10.1609/icaps.v32i1.19843.
[31] M. Jaeger, P.G. Jensen, K.G. Larsen, A. Legay, S. Sedwards, J.H. Taankvist, Teaching stratego to play ball: Optimal synthesis for continuous space MDPs, in: Automated Technology for Verification and Analysis, in: Lecture Notes in Computer Science, 2019, pp. 81–97, http://dx.doi.org/10.1007/978-3-030-31784-3_5.
[32] E.F. Camacho, C.B. Alba, Model Predictive Control, Springer, 2013.
[33] M.A. Goorden, K.G. Larsen, J.E. Nielsen, T.D. Nielsen, M.R. Rasmussen, J. Srba, Learning safe and optimal control strategies for storm water detention ponds, IFAC-PapersOnLine 54 (5) (2021) 13–18, http://dx.doi.org/10.1016/j.ifacol.2021.08.467.
[34] W.C. Huber, L.A. Rossman, R.E. Dickinson, EPA storm water management model, SWMM5, Watershed Models 338 (2005) 359.
[35] A.T.H. Thomsen, J.E. Nielsen, M.R. Rasmussen, A simplified method for measuring the discharge from stormwater detention ponds, 2019, (in preparation).
[36] P. Ashok, J. Křetínský, K.G. Larsen, A. Le Coënt, J.H. Taankvist, M. Weininger, SOS: Safe, optimal and small strategies for hybrid Markov decision processes, in: Quantitative Evaluation of Systems, in: Lecture Notes in Computer Science, 2019, pp. 147–164, http://dx.doi.org/10.1007/978-3-030-30281-8_9.
[37] M.L. Puterman, Markov Decision Processes: Discrete Stochastic Dynamic Programming, John Wiley & Sons, 1994.
[38] G. Behrmann, A. David, K.G. Larsen, A tutorial on Uppaal, in: Formal Methods for the Design of Real-Time Systems, in: Lecture Notes in Computer Science, 2004, pp. 200–236, http://dx.doi.org/10.1007/978-3-540-30080-9_7.
[39] P.E. Bulychev, A. David, K.G. Larsen, M. Mikučionis, D.B. Poulsen, A. Legay, Z. Wang, Uppaal-SMC: statistical model checking for priced timed automata, in: Quantitative Aspects of Programming Languages, (85) EPTCS, 2012, pp. 1–16, http://dx.doi.org/10.4204/EPTCS.85.1.
[40] G. Behrmann, A. Cougnard, A. David, E. Fleury, K.G. Larsen, D. Lime, Uppaal-tiga: Time for playing games!, in: International Conference on Computer Aided Verification, in: Lecture Notes in Computer Science, 2007, pp. 121–125, http://dx.doi.org/10.1007/978-3-540-73368-3_14.
[41] M.A. Goorden, K.G. Larsen, J.E. Nielsen, T.D. Nielsen, W. Qian, M.R. Rasmussen, J. Srba, G. Zhao, Models for "Optimal Control Strategies for Stormwater Detention Ponds", Zenodo, 2022, http://dx.doi.org/10.5281/zenodo.6592380.
[42] Danish Meteorological Institute, DMI homepage, 2020, URL dmi.dk.
[43] S.A. Nielsen, E. Hansen, Numerical simulation of the rainfall-runoff process on a daily basis, Hydrol. Res. 4 (3) (1973) 171–190, http://dx.doi.org/10.2166/nh.1973.0013.
[44] B.E. McDonnell, K. Ratliff, M.E. Tryby, J.J.X. Wu, A. Mullapudi, PySWMM: The Python interface to stormwater management model (SWMM), J. Open Source Softw. 5 (52) (2020) 2292, http://dx.doi.org/10.21105/joss.02292.
[45] M.A. Goorden, P.G. Jensen, K.G. Larsen, M. Samusev, J. Srba, G. Zhao, STOMPC: Stochastic model-predictive control with Uppaal Stratego, in: A. Bouajjani, L.s. Holí k, Z. Wu (Eds.), Automated Technology for Verification and Analysis, in: Lecture Notes in Computer Science, Springer International Publishing, 2022, pp. 327–333, http://dx.doi.org/10.1007/978-3-031-19992-9_21.
[46] L.A. Rossman, Storm water management model user's manual version 5.1, 2015, URL https://www.epa.gov/sites/default/files/2019-02/documents/epaswmm5_1_manual_master_8-2-15.pdf.