Distributed tree-based model predictive control on a drainage water system

J. M. Maestre, L. Raso, P. J. van Overloop and B. De Schutter

ABSTRACT

Open water systems are one of the most externally influenced systems due to their size and continuous exposure to uncertain meteorological forces. The control of systems under uncertainty is, in general, a challenging problem. In this paper, we use a stochastic programming approach to control a drainage system in which the weather forecast is modeled as a disturbance tree. Each branch of the tree corresponds to a possible disturbance realization and has a certain probability associated to it. A model predictive controller is used to optimize the expected value of the system variables taking into account the disturbance tree. This technique, tree-based model predictive control (TBMPC), is solved in a distributed fashion. In particular, we apply dual decomposition to get an optimization problem that can be solved by different agents in parallel. In addition, different possibilities are considered in order to reduce the communicational burden of the distributed algorithm without reducing the performance of the controller significantly. Finally, the performance of this technique is compared with others such as minmax or multiple MPC.

Key words | control of water systems, distributed model predictive control, stochastic programming

J. M. Maestre (corresponding author)
Systems and Automation Department,
University of Seville,
Spain
E-mail: *pepemaestre@us.es*

L. Raso
P. J. van Overloop
Department of Water Management,
Delft University of Technology,
The Netherlands

B. De Schutter
Delft Center for Systems and Control,
Delft University of Technology,
The Netherlands

INTRODUCTION

Societies living near open waters strive at managing these waters by trying to control the water levels in them with structures such as gates and pumps. Different control methods have been used for this over the last few decades, such as feedforward control (Ahn 2000), feedback control (Clemmens & Schuurmans 2004), or model predictive control (MPC) (van Overloop 2006). Other important algorithms detailed in the literature and an introduction to basic control concepts regarding the control of irrigation canals can be found in Malaterre *et al.* (1998). In general, the performance of feedback controllers is reduced by the transport delays that are usually present in this kind of system. The use of feedforward controllers solves the problem regarding the delays, but unfortunately these controllers are not able to cope with the constraints involved in these control problems, e.g. maximum or minimum flow capacities. The only technique that allows us to handle in a systematic manner multi-variable interactions, constraints on manipulated inputs and system states, and optimization requirements is MPC. The future predictions of the state, output, and input variables are generated along a given prediction horizon using a mathematical model of the system and are used to minimize a given performance index, which is a cost function that defines the optimization criterion used to determine the best possible control action sequence. Due to its versatility, MPC has become very popular in the industry, and many implementations can be found (Camacho & Bordons 2004). Likewise, MPC has been used or proposed many times for the control of water systems. For example, in Zafra-Cabeza *et al.* (2011), a risk minimization strategy for the control of irrigation canals is proposed. In Gomez *et al.* (2002), a decentralized predictive control strategy is considered. Finally, in van Overloop *et al.* (2010), a real-time implementation of MPC is described.

As they are open environmental systems, open waters are vulnerable to meteorological influences. Due to recent advances and the availability of historical data, these

doi: 10.2166/hydro.2012.125

disturbances on the system can be predicted better and over longer horizons. For this reason, in this paper, we are especially interested in the way MPC deals with uncertainty. The simplest method is to ignore it and let the controller work in a nominal and deterministic fashion, which often results in poor control performance. A more sophisticated way to proceed is to take the uncertainties explicitly into account, e.g. using deterministic forecasts if the certainty equivalence property holds (van de Water & Willems 2002), or assuming a bounded set of unmeasurable disturbances in order to use a minmax approach (Witsenhausen 1968). Unfortunately, the certainty equivalence property does not hold for meteorological disturbances (they are not distributed in a Gaussian manner), and minmax controllers have shown to be slow and too conservative (Muñoz de la Peña *et al*. 2005; de la Peña *et al*. 2007). This leads to alternative approaches to deal with uncertainty, such as stochastic programming (SP) (Muñoz de la Peña *et al*. 2005), which models unknown disturbances as random variables and focuses on the control of the expected value of the system variables while guaranteeing robust constraint satisfaction. Under this approach, unknown disturbances can be also taken into account by considering different representative realizations of the disturbances. For example, this idea has been previously used for the control of water systems in van Overloop *et al*. (2008), where multiple MPC (MMPC) is proposed. MMPC calculates the control input sequence that minimizes a given performance index for different deterministic disturbance scenarios. The probability of occurrence of each scenario is taken into account, so that the most likely scenarios have greater impact on the performance index, i.e. their influence on the optimal control sequence is higher.

In this work, we use tree-based model predictive control (TBMPC), which is an SP-MPC scheme that assumes that the time evolution of the most relevant possible disturbance signals can be synthesized in a rooted tree (Mulvey & Ruszczynski 1995; Raso *et al*. 2009). Each root-to-leaf path is a possible disturbance scenario, i.e. branches appear in the tree as the different disturbance forecasts diverge along the prediction horizon. Given that each node of the tree corresponds to a point in time in which a control action can be implemented, control sequences are not allowed to diverge before disturbance sequences. Thus, it is necessary to introduce compatibility constraints (some authors also refer to these constraints as non-anticipativity constraints) (Rockafellar & Wets 1991) in order to limit the anticipative nature of the MPC controller. The role of these constraints is simply to force the controller to calculate control trajectories able to cope with all the scenarios until a bifurcation point allows us to discard the realization of some of them. As a result of this, the controller provides a rooted tree of control actions that are calculated according to the different sequences in the disturbance tree.

At this point, it is convenient to point out that the computational burden of TBMPC may become an important issue. All studies mention the so-called 'curse of dimensionality' as the main drawback that comes with optimization problems, especially when treated stochastically. Moreover, water management control problems usually involve long control horizons, which hinders the application of MPC for large-scale systems (Negenborn *et al*. 2006). In order to overcome this problem, different distributed MPC (DMPC) techniques have been proposed during the last decade; see Scattolini (2009) for an extensive survey on this topic. In general, DMPC focuses on the application of MPC to systems that are composed of several subsystems governed by local controllers or *agents*. Each of these agents implements a local MPC controller and may or may not share information with the other subsystems. The distribution of a centralized problem between several agents allows us to solve problems that would be intractable otherwise. In this paper, we will use dual decomposition (Negenborn *et al*. 2008; Rantzer 2009) in order to distribute the TBMPC optimization problem. This approach has also been previously used in water management, in particular, for water resource system planning; see Escudero (2000). In particular, the distribution will be done according to the different scenarios: we assume that each possible trajectory contained in the disturbance tree is assigned to an agent. Therefore, the number of agents is the same as the number of terminal vertexes contained in the tree. Each of these agents optimizes the same performance index assuming its corresponding disturbance sequence is known and deterministic. In addition, each agent must respect the compatibility constraints, i.e. each

agent must reach a consensus on the value of its control sequence with those agents whose disturbance sequences follow the same trajectory. Note that all the agents must agree on the value of the first control step, the one applied to the system in the closed-loop receding horizon. In Figure 1, the proposed scheme is shown. For simplicity, only three different scenarios for the disturbance signals are considered. Then, each of these scenarios is assigned to a different agent, which computes the optimal control policy for the particular scenario while taking into account the control goal and the constraints of the problem. Note that compatibility constraints require agents to choose the same value for their control actions at the time step in which they are active. The main contribution of this paper is the novel application of distributed control in order to consider different scenarios in the control of water systems. To the best of our knowledge, only Escudero (2000) followed a similar approach, but it was in the context of operation research and water resource planning, and hence not control. Although the distributed control technique that we use, dual decomposition, is very standard and has been previously used in the context of irrigation canals (Negenborn *et al.* 2009), we also show in this paper ways to reduce its communicational burden. Specifically, we show that the limitation of the compatibility constraints provides similar closed-loop performance with significantly less communicational burden. Likewise, we study different thresholds as stopping conditions in the parallel algorithm. Finally, the last contribution of the paper is a comparison of the performance of the proposed controller and other techniques used in the literature.

METHODS

Ensemble forecasting

In open water systems, the uncertainty is generally introduced by the unpredictable nature of the weather. Specifically, runoff derived from rainfall is the major source

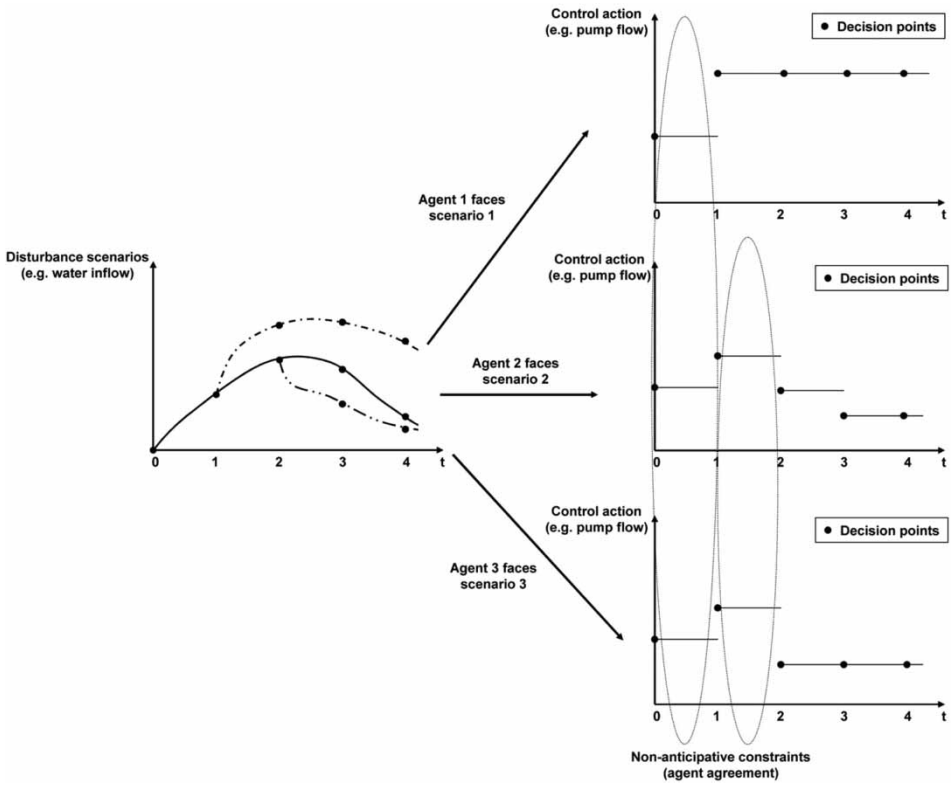

Figure 1 | Proposed control scheme.

of uncertainty in this context. An ensemble forecast (EF) is a weather prediction composed of possible trajectories of its evolution. It is generated by a model that is run using different initial conditions or different numerical representations of the atmosphere, accounting for the major sources of forecast uncertainty (Gneiting & Raftery 2005). These trajectories generally have small differences at the initial stage of the forecast; then they tend to diverge because of the chaotic nature of the underlying model. Finally, notice that each of the trajectories in the EF has a certain associated probability.

For simulation purposes, a large number of scenarios improves the accuracy of the stochastic approach. However, this may be troublesome as it may lead to an excessive growth of the computational burden. In order to avoid this problem, a *representative* subset of scenarios can be chosen. In order to provide the controller with the set of the most representative ensemble members, a scenario reduction algorithm is applied (Growe-Kuska *et al.* 2003). This reduced ensemble is bundled into a tree that will allow us to set up a multistage SP problem. A tree, differently from an ensemble, specifies when the trajectories diverge from each other. This moment in time is called the bifurcation point. At any bifurcation point, the space of ensemble members is divided into two subsets, mutually exclusive. In general, the tree is generated from the ensemble by aggregating trajectories over time until the difference between them becomes such that they can be no longer assumed to be similar. At such a point, a bifurcation is produced and the tree branches. The operation of tree generation has been the subject of numerous studies, for example, Growe-Kuska *et al.* (2003), Sutiene *et al.* (2010) and Raso *et al.* (2012).

At this point, it is important to remark that in this paper, we assume that all the scenarios contained in a tree have the same value at the first step of the prediction horizon. This assumption is reasonable in the context of open water systems, where the weather forecasts provide a good indication about how the disturbances will be, especially at the beginning of the forecast period.

MPC

An MPC controller assumes the system dynamics can be modeled mathematically and uses this knowledge to calculate what the optimal control actions are according to a given cost function. In this paper, we will assume that the model of the system can be described by the following discrete-time equation:

$$x(k+1) = Ax(k) + Bu(k) + Ew(k) \tag{1}$$

where $x(k) \in \mathbb{R}^q$ is the state of the system at time step k, $u \in \mathbb{R}^r$ is the vector of manipulated variables, and $w \in \mathbb{R}^s$ is a vector of measurable disturbances. A, B, and E are matrices of proper dimensions. We consider the following linear constraints in the states and the inputs:

$$\begin{aligned} x &\in \mathcal{X} \\ u &\in \mathcal{U} \end{aligned} \tag{2}$$

where $\mathcal{X}$ and $\mathcal{U}$ are closed polyhedra defined by a system of linear inequalities. As we will see later, this basic model has enough accuracy to provide us with a reasonable model for the control of some water systems. We will assume that our control goal is the regulation of the state vector to a given value. For example, this corresponds to the situation in which we want to regulate the water level to the value of a given reference. Without loss of generality, we can assume that the control objective consists of regulating the state vector to the origin (otherwise a simple change of variable takes us to this case). It is therefore possible to define the following performance index:

$$\begin{aligned} J(U, x_0) = &\sum_{k=0}^{N_h-1} \left(x^T(k)Qx(k) + Q_l x(k) + u^T(k)Ru(k)\right) \\ &+ x^T(N_h)Qx(N_h) + Q_l x(N_h) \end{aligned} \tag{3}$$

where Q, Q_l, and R are constant weighting matrices of the proper size and N_h is the prediction horizon. Note that with this performance index, we penalize quadratically and linearly the deviation of the state vector with respect to the origin, and also quadratically the value of the manipulated variables with respect to zero. These penalties are calculated and summed over the next N_h time samples. Taking into account (1), it can be seen that function J depends on the control input sequence $U = (u(0), \ldots, u(N_h - 1))$ and the value of the state at time step $k = 0$, x_0. The MPC controller calculates the

optimal control action by minimizing this cost:

$$
\begin{aligned}
&U^* = \underset{U}{\operatorname{argmin}} J(U, x_0)\\
&\text{s.t.}\\
&x(k+1) = Ax(k) + Bu(k) + Ew(k)\\
&x(k) \in \mathcal{X}\ \forall k \in \{1,\ \ldots, N_{\mathrm{h}}\}\\
&u(k) \in \mathcal{U}\ \forall k \in \{0,\ \ldots, N_{\mathrm{h}} - 1\}\\
&x(0) = x_0\\
&w(k) = w_k
\end{aligned}
\tag{4}
$$

where $w_k = (w_0, w_1, w_2,\ \ldots, w_{N_{\mathrm{h}}})$ is a deterministic sequence composed of the present and future values of the disturbances. The result of this optimization problem is the optimal control sequence $U^* = (u^*(0),\ \ldots, u^*(N_{\mathrm{h}} - 1))$. Only the first component of U^* is applied. The rest of the sequence provides information about the expected evolution of the manipulated variables in the future, but is not implemented during the next sample times. At the next sample instant, the optimization problem (4) is solved using the current state at that time and the most recent disturbance forecast. The first component of the resulting control sequence is implemented again. This procedure is repeated every sample time in what is usually denominated a receding horizon strategy.

Distributed TBMPC uses the tree provided by the ensemble forecasting and solves (4), taking into account the sequences contained in the tree. In particular, problem (4) is solved for each possible disturbance sequence in the tree. Notice that the resulting set of problems can be solved by a set of agents in parallel. Thus, if there are N_{s} different scenarios in the tree, there are N_{s} different agents working in parallel. Nevertheless, the problem faced by the agents has more constraints than (4). Additional restrictions have to be imposed in order to account for the compatibility in the optimization procedure. Let us therefore define the binary auxiliary function $\delta_{i,j}(k)$ in the following way:

- $\delta_{i,j}(k)$ is equal to 1 if the control input of agent i must have the same value of the control input of agent j at time k. That is, $\delta_{i,j}(k) = 1$ iff $u_i(k) = u_j(k)$.
- $\delta_{i,j}(k)$ is zero otherwise.

In order to have an input tree with the same structure as the disturbance tree, $\delta_{i,j}(k)$ must be 1 for $k = 0,\ \ldots, k_{ij}$, where k_{ij} is the first time step at which the disturbance sequences corresponding to agents i and j are different. From a centralized point of view, the problem solved by a TBMPC controller is the following:

$$
\begin{aligned}
&\min_{U_1,\ldots,U_{N_{\mathrm{s}}}} \sum_{i=1}^{N_{\mathrm{s}}} \alpha_i J(U_i, x_0)\\
&\text{s.t.}\\
&\left.\begin{aligned}
&x_i(k+1) = Ax_i(k) + Bu_i(k) + Ew_i(k)\\
&x_i(k) \in \mathcal{X}\ \forall k \in \{1,\ \ldots, N_{\mathrm{h}}\}\\
&u_i(k) \in \mathcal{U}\ \forall k \in \{0,\ \ldots, N_{\mathrm{h}} - 1\}\\
&x_i(0) = x_0\\
&w_i(k) = w_{i,k}\\
&\delta_{i,j}(k)u_i(k) = \delta_{i,j}(k)u_j(k)\ \forall i,j \in \{1,\ldots,N_{\mathrm{s}}\},\ k \in \{0,\ \ldots, N_{\mathrm{h}} - 1\}
\end{aligned}\right\}\\
&\quad \forall i \in \{1,\ \ldots, N_{\mathrm{s}}\}
\end{aligned}
\tag{5}
$$

where $w_{i,k} = (w_{i,0}, w_{i,1},\ \ldots, w_{i,N_{\mathrm{h}}})$ is a deterministic sequence composed of the present and future values of the disturbances faced by agent i, and α_i is the probability assigned to the event that the disturbance sequence $w_{i,k}$ actually occurs.

Dual decomposition

In order to solve (5) in a distributed fashion, it is necessary to remove the coupled constraints, which in this case are given by equalities of the type $u_i(k) = u_j(k)$. Dual decomposition can be used for this, that is, to distribute the centralized problem (5) between the agents. The introduction of Lagrange multipliers $\lambda_{i,j}(k)$ in the cost function allows us to separate the problem. In particular, it can be proven that solving (5) is equivalent to solving the following optimization problem (Boyd & Vandenberghe 2004):

$$
\begin{aligned}
&\max_{\Lambda} \min_{U_1,\ldots,U_{N_{\mathrm{s}}}}\\
&\quad \times \sum_{i=1}^{N_{\mathrm{s}}} \left(\alpha_i J(U_i, x_{i,0}) + \sum_{j=1}^{N_{\mathrm{s}}} \sum_{k=0}^{N_{\mathrm{a}}} \delta_{i,j}(k)\lambda_{i,j}(k)(u_i(k) - u_j(k)) \right)\\
&\text{s.t.}\\
&\left.\begin{aligned}
&x_i(k+1) = Ax_i(k) + Bu_i(k) + Ew_i(k)\\
&x_i(k) \in \mathcal{X}\ \forall k \in \{1,\ \ldots, N_{\mathrm{h}}\}\\
&u_i(k) \in \mathcal{U}_i\ \forall k \in \{0,\ \ldots, N_{\mathrm{h}} - 1\}\\
&x_i(0) = x_0\\
&w_i(k) = w_{i,k}
\end{aligned}\right\} \forall i \in \{0,\ \ldots, N_{\mathrm{s}}\}
\end{aligned}
\tag{6}
$$

where $\Lambda = \{\lambda_{i,j}(k),\ \forall i, j \in \{1, .., N_s\},\ k \in \{0, N_a\} | \delta_{i,j}(k) = 1\}$ is the set of all the Lagrange multipliers, sometimes referred to as a set of prices. Notice that we have introduced a new parameter, N_a, which is the agreement horizon. Ideally, $N_a = N_h - 1$, i.e. the agents have to respect the compatibility constraints defined over the entire horizon. Nevertheless, it may be desirable to limit the effect of these constraints in time so that the number of coupling constraints is reduced.

As can be seen, problem (6) can be solved in a distributed fashion. The following algorithm shows the distributed optimization procedure that takes place at time step k.

- Step 0: let l be the index used to count the number of iterations of the procedure. Initially, $l = 0$ and an initial set of prices Λ^0 is given.
- Step 1: at each iteration l, each agent i calculates its own optimal input trajectory solving the following problem for a particular set of values of the Lagrange multipliers Λ^l:

$$U_i^* = \underset{U_i}{\operatorname{argmin}} \left(\alpha_i J(U_i, x_0) + \sum_{k=0}^{N_a} u_i(k) \sum_{j \neq i} \delta_{i,j}(k) \lambda_{i,j}^l(k) \right)$$

s.t.

$$\begin{aligned} & x_i(k+1) = Ax_i(k) + Bu_i(k) + Ew_i(k) \\ & x_i(k) \in \mathcal{X}\ \forall k \in \{1, \ldots, N_h\} \\ & u_i(k) \in \mathcal{U}\ \forall k \in \{0, \ldots, N_h - 1\} \\ & x_i(0) = x_0 \\ & w_i(k) = w_{i,k} \end{aligned} \tag{7}$$

Notice that (7) corresponds to the part of (6) that corresponds to agent i.

- Step 2: once the input trajectories have been calculated, the prices of agent i are updated by a gradient step as follows:

$$\lambda_{i,j}^{l+1}(k) = \lambda_{i,j}^l(k) + \gamma^l (u_i(k) - u_j(k)) \tag{8}$$

Notice that a price is only changed whenever there is a disagreement about the value of the variable between the agents. Once the agents agree on a value, the price remains constant. Convergence of these gradient algorithms has been proven under different types of assumptions on the step size sequence γ^l. See, for example, Shor (1985). Note that in order to update the prices, the agents must communicate. See Maestre *et al.* (2010) for alternatives in the way the prices can be updated.

- Step 3: let $\Delta u(k) = \max_{i,j} |u_i(k) - u_j(k)|$. The algorithm stops if $\Delta u(k) \leq \Delta u_{max}$, where Δu_{max} is a parameter that represents the maximum allowable difference between the proposals of any two agents. In case $u(k)$ is a vector, this criterion is applied component wise. Alternatively, the algorithm also stops if the number of iterations l exceeds a given threshold l_{max}. Otherwise, the process is repeated from step 1 for $l = l + 1$.

Remark. Notice that the stopping criterion that we use in this paper is similar to the establishment of a threshold on the price increment. Other criteria are possible, though. For example, in Giselsson & Rantzer (2010), it is proved how, under certain assumptions, it is possible to establish suboptimality bounds on this iteration procedure.

Remark. Strictly speaking, the term *parallel* should be used when the control problem is solved locally, e.g. by a multicore computer or cluster of computers, and the term *distributed* when there are computers at different locations cooperating to obtain a joint solution. Throughout this paper, both terms are used indistinguishably because dual decomposition can be used either way. Nevertheless, the simulations we show in the next section are solved in a *parallel* fashion.

RESULTS AND DISCUSSION

We have applied the distributed TBMPC controller to a model of a real drainage system described in van Overloop *et al.* (2008). In Figure 2, a sketch of the water system can be seen. There are three important variables in this figure. In the first place, we have the controlled variable, h (m), which is the average water level with respect to the average sea level. The second variable is Q_c (m^3/s), which represents the effect of pumping water out of the system and is the manipulated variable. Note that there is a constraint present in this actuator, which is the maximum pump capacity. Finally, there is a disturbance term given by Q_d (m^3/s), which stands for the inflow of water due to rainfall. The discrete-time model used to represent the dynamics of the drainage canal system is:

$$h(k+1) = h(k) - \frac{T_c}{A_s} Q_c(k - k_d) + \frac{T_c}{A_s} Q_d(k) \tag{9}$$

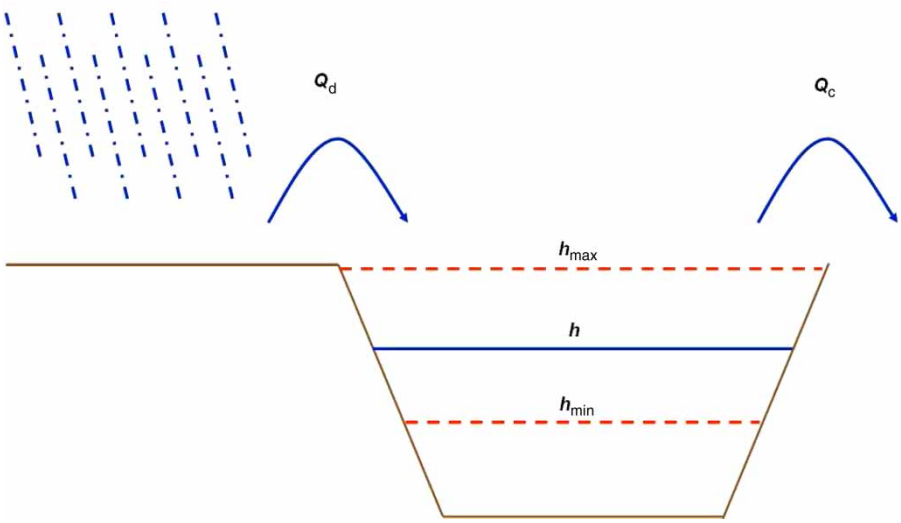

Figure 2 | Schematization of drainage water system.

where A_s is the average storage area (m^2), T_c is the control time step (s), k_d is the number of delay steps between control action and change in average water level, and k is the time step index. A state space model based on (9) will be used for the controller. In particular, the model will focus on the error between the current water level and the water level setpoint. Let h_{ref} be the constant water level setpoint. Thus, the error $e(k)$ in the water level can be calculated as $e(k) = h(k) - h_{ref}$. In addition, the state vector will be expanded with the state variable $e_{zone}(k)$ in order to represent explicitly how much the error $e(k)$ is above or below a given safety margin defined around the water level setpoint. Taking all of this into consideration, we can define the following space state model:

$$\begin{bmatrix} e(k+1) \\ e_{zone}(k) \\ Q_c(k) \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\frac{T_c}{A_s} \\ 1 & 0 & -\frac{T_c}{A_s} \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} e(k) \\ e_{zone}(k-1) \\ Q_c(k-1) \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & -1 \\ 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \Delta Q_c(k) \\ u_{zone}(k) \end{bmatrix} + \begin{bmatrix} \frac{T_c}{A_s} \\ \frac{T_c}{A_s} \\ 0 \end{bmatrix} \cdot [Q_d(k)] \quad (10)$$

Note that a new variable has been introduced in this model, $u_{zone}(k)$, which is related to the constraints of the optimization problem the MPC controller solves. In particular, hard constraints are defined on the input of the system and soft constraints are introduced on the state:

$$Q_c(k) \geq 0$$
$$Q_c(k) \leq Q_{c,max}$$
$$u_{zone}(k) \geq h_{min} - h_{ref}$$
$$u_{zone}(k) \leq h_{max} - h_{ref}$$

These constraints help us explain better the meaning of $u_{zone}(k)$. According to (10), $e_{zone}(k)$ may seem equal to a duplicated version of $e(k+1)$. The difference between them is that $e_{zone}(k)$ is affected by the auxiliary manipulated variable $u_{zone}(k)$, which is able to compensate this error as long as the water level stays inside the region defined by h_{min} and h_{max}. Therefore, note that $u_{zone}(k)$ has no physical meaning and it is introduced only in order to avoid the possible infeasibility problems that could have been derived from the use of hard constraints on the state. In addition, note that $e_{zone}(k)$ will be zero as long as $e(k+1)$ stays between h_{min} and h_{max}.

As has been stated, there is a disturbance in (10), which is the inflow due to rainfall $Q_d(k)$. This inflow consists of the drainage and the pumping of the water precipitated into the polder area. The future disturbance behavior is modeled as a tree that contains its most representative possible trajectories along the prediction horizon. The data are contained in the tree that is given to the controller each time sample k is obtained from artificial EFs that are generated using

meteorological models. These ensembles are generated modeling the rainfall event R_t at time t as the product of two random variables $R_t = B_t \times E_t$, where B_t is a binary random variable representing the occurrence of the rain event and E_t is a gamma distributed random variable representing the quantity of precipitation. The mean and variance of E_t are determined by γ_t, which is related to the uncertainty level so that:

$$E_t \sim \Gamma\begin{cases} \mu_t = (1-\gamma_t)P_t + \gamma_t \mu_{\mathrm{cl}} \\ \sigma_t^2 = \gamma_t \cdot \sigma_{\mathrm{cl}}^2 \end{cases} \tag{11}$$

where μ_{cl} and σ_{cl}^2 are the climatic mean and variance of the amount of precipitation, which are meteorological parameters, and P_t is the observed precipitation at time t. The uncertainty along the forecasting horizon varies according to $\gamma_t = t/N_{\mathrm{h}}$ for $t \leq N_{\mathrm{h}}$, where N_{h} determines the information prediction horizon (Weijs *et al.* 2007). This model allows us to generate artificially a set of 80 different scenarios that are later condensed into a disturbance tree.

In Figures 3(a)–3(d), we show the results of a closed-loop simulation of the proposed scheme over 25 hours, which corresponds to 100 time steps. As can be seen, the simulation takes place during a stormy event that tests the controller capability to keep the water level within the desired margin. The numerical values of the parameters that characterize the system and the cost function of the controller can be seen in Table 1. In addition, we have

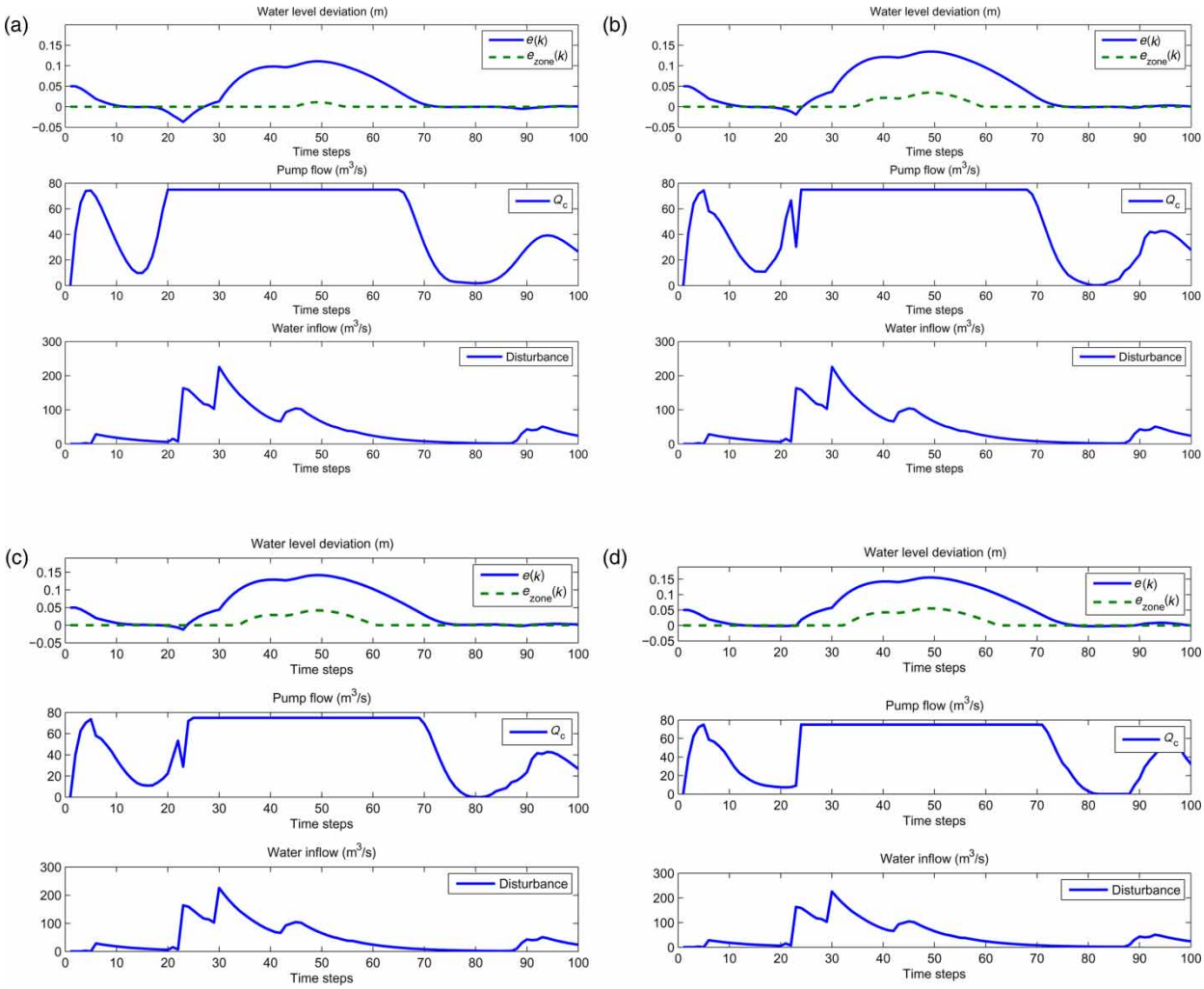

Figure 3 | Simulation cases: (a) centralized TBMPC, (b) distributed TBMPC, (c) distributed TBMPC with $N_a = 1$, (d) minmax.

Table 1 | Model and controller parameters

Parameter	Symbol	Value
Storage area	A_s	7,300,000 (m^2)
Control time step	T_c	900 (s)
Number of scenarios in the tree	N_s	6
Prediction horizon	N_h	16 (4 hours)
Control horizon	N_c	16 (4 hours)
Maximum pump capacity	Q_{max}	75 (m^3/s)
Delay in the manipulated variable	k_d	1
Maximum number of iterations allowed	l_{max}	2,000
Agreement horizon	N_a	10
Disagreement threshold	Δu_{max}	1 (m^3/s)
Quadratic penalty weight on e	Q_e	180
Quadratic penalty weight on e_{zone}	$Q_{e_{zone}}$	180×10^5
Linear penalty weight on Q_c	Q_{l,Q_c}	130×10^{-5}
Quadratic penalty weight on ΔQ_c	$R_{\Delta Q_c}$	10^{-6}
Quadratic penalty weight on u_{zone}	$R_{u_{zone}}$	10^{-16}

carried out simulations with values different from the ones shown in Table 1 in order to see their influence on the controller performance. In particular, these additional simulations were carried out for different values of the agreement horizon, N_a, and the disagreement threshold, Δu_{max}.

At first, in Figure 3(a), the results of a centralized MPC controller with a perfect forecast are depicted. Notice that this case is given only as a reference since a perfect forecast, i.e. an exact knowledge about the future evolution of weather, cannot be obtained. However, this case gives us an upper bound for the performance of the rest of the controllers. In Figure 3(b), the centralized TBMPC configured with the parameters of Table 1 can be seen. Figure 3(c) shows the results of the TBMPC strategy when the agreement horizon $N_a = 1$. It is remarkable that the approximation does not lead to significantly different results. Figure 3(d) shows the results corresponding to the use of the minmax controller. Finally, the plots corresponding to the probabilistic MPC or MMPC are not included because they are very similar to 3(b) for this particular case.

In general, it is not possible to calculate the exact number of optimization variables that the centralized TBMPC problem has; it depends completely on the structure of the tree. Nevertheless, it is possible to calculate an upper and a lower bound for this number of variables. The upper bound of the number of optimization variables is $r \times N_s \times (N_c - 1) + r$, where r is the dimension of the manipulated variables vector, N_s is the number of scenarios and N_c is the control horizon. Notice that this bound is calculated for a case in which the tree branches completely after the first time step in the horizon. The lower bound is given simply by $r \times N_c$, assuming that there is only one scenario in the tree. It is obvious that as the number of scenarios grows, so does the number of optimization variables. At some point, the number of variables may become intractable from a computational point of view. Here is a reason that supports the use of distributed algorithms, but there are more. For example, the system could be naturally distributed, such as a large water network in which there are several areas controlled by different local controllers. Without loss of generality, we can assume that each node faces a problem similar to (4). Given that the overall control problem comprises all the nodes' subproblems, the total number of optimization variables grows with the number of nodes in the network. Again, at some point, there may be computational problems due to an excessive number of optimization variables. Another suitable situation for the use of a distributed algorithm would be a multi-objective optimization in which several agents with different goals try to reach an agreement about the use of a common resource. The distributed algorithm that we use in this paper substitutes a problem with up to $r \times N_s \times (N_c - 1) + 1$ optimization variables by N_s separable problems with $r \times N_c$ variables that have to be solved iteratively until an agreement has been reached. The size of the problem that we use in this paper as an example is not big enough to make the use of parallel computation indispensable, i.e. it is not hard to solve the problem in a centralized fashion. Nevertheless, the use of parallel computation here helps us to illustrate how the problem is distributed and the iterative procedure that takes place for the convergence. The number of optimization variables of our example will be bounded between 16 and 182. Its distributed version is composed of six problems with 32 optimization variables for each one.

As has been stated in the previous paragraph, the separation of the original optimization problem into several

separable optimization problem comes at a price: these problems have to be solved iteratively until an agreement has been obtained. Hence, one of the most important aspects in the distributed control scheme is the number of iterations that are necessary for convergence. In Table 2, the number of iterations needed for convergence is shown for different values of the agreement horizon, N_a. As expected, a greater value of N_a implies a higher number of iterations, which is logical since it implies a higher number of variables on which the agents must reach an agreement. A question that arises naturally is how a change of N_a affects the performance of the controller. Table 2 also helps to answer this question. The pump flow difference between the centralized TBMPC and its distributed versions has been analyzed in order to calculate its maximum, mean, and standard deviation along the 100 time steps of the simulation. As can be seen, the results are quite similar. Actually, the maximum deviation from the control signal provided by the centralized TBMPC is always below 3%. These results suggest that the completeness of the set of compatibility constraints is not so relevant in the calculation of the first component of the control vector. In addition, Table 2 shows, in its last column, the value of the cumulated cost of the closed-loop system at the end of the simulation. This value is calculated according to the performance index defined in (3) and the parameters given in Table 1. It is interesting to observe how the performance of these controllers was very similar to the performance shown by the centralized TBMPC. As can be seen, the number of iterations can be dramatically reduced while obtaining a very similar performance by means of a proper use of the agreement horizon N_a.

Remark. All the distributed versions of the centralized TBMPC problem include simplifications with respect to the original problem introduced by the convergence condition $\Delta u(k) \leq \Delta u_{max}$ and the elimination of the compatibility constraints beyond the agreement horizon N_a. While these simplifications explain the slight loss of performance of these alternatives, in general, it is not possible to quantify exactly the loss amount for different simulations. Moreover, Table 3 shows that alternatives theoretically closer to the original problem can show worse performance than others that include more simplifications.

Remark. Notice that this fact does not question the rationale behind the disturbance tree. Actually, the compatibility constraints allow us to see a tree of the most likely control signals in the future. Without them, the controller always anticipates, and therefore the future values of the control signals are not as realistic as possible.

Table 2 shows that the maximum deviation from the control signal provided by the centralized TBMPC is always below 3%, but this value can be reduced with a proper adjustment of the disagreement threshold Δu_{max}, which is another degree of freedom with a strong influence on the number of iterations. For this reason, Table 3 shows the impact of a variation of the disagreement threshold Δu_{max}. It can be seen that as Δu_{max} grows, the number of iterations for convergence and the quality of the solution decrease.

Remark. Notice that we have not included figures showing the simulations with the distributed versions of the TBMPC controller. The reason is that the differences are

Table 2 | Impact analysis of an agreement horizon variation ($\Delta u_{max} = 1$)

	Iterations / for convergence			Pump flow difference (m^3/s)			
N_a	Max	Mean	Std. dev.	Max	Mean	Std. dev.	Cum. cost
Cen.	–	–	–	0	0	0	2.31×10^5
1	119	16.56	27.44	2.35	0.00	0.58	2.44×10^5
3	263	38.17	64.49	2.03	– 0.01	0.51	2.35×10^5
5	565	70.84	121.57	1.19	– 0.02	0.35	2.37×10^5
7	813	109.69	182.79	0.97	– 0.02	0.35	2.40×10^5
9	1,009	147.81	240.88	1.34	– 0.02	0.41	2.35×10^5
16	1,321	169.82	279.41	1.09	– 0.03	0.39	2.32×10^5

Table 3 | Impact analysis of a disagreement threshold Δu_{max} variation ($N_a = 16$)

	Iterations *I* for convergence			Pump flow difference (m^3/s)			
Δu_{max}	Max	Mean	Std. dev.	Max	Mean	Std. dev.	Cum. cost
Cen.	–	–	–	0	0	0	2.31×10^5
0.25	2,720	460.95	681.53	0.24	0.00	0.09	2.32×10^5
0.5	1,964	297	461.25	0.35	0.00	0.12	2.32×10^5
1	1,321	169.82	279.41	1.09	−0.03	0.39	2.32×10^5
1.5	945	108.86	193.99	1.53	−0.03	0.48	2.38×10^5
2	680	79.15	141.83	1.86	0.00	0.63	2.42×10^5

negligible when these figures are compared with Figure 3(b). Tables 2 and 3 clearly point out the similarities between the centralized and distributed solutions.

Finally, we have performed closed-loop simulations with several controllers for the sake of comparison. Specifically, we have tested the following controllers:

- PFMPC: centralized MPC with perfect forecast.
- TBMPC: centralized TBMPC controller.
- DTBMPC*x*: distributed implementation of the TBMPC controller where *x* is the value of the agreement horizon. The following values were tested: $N_a = 1, 3, 5, 7, 9, 16$.
- ProbMPC: centralized MPC with a single disturbance forecast consisting of the weighted average of all the scenarios contained in the tree based on their probability.
- MMPC: multiple MPC, which is presented in Overloop *et al.* (2008). Basically, it is an MPC controller that calculates the control vector that minimizes several scenarios at the same time. In this case, we have used this methodology with all the scenarios and corresponding probabilities contained in the tree.
- MinmaxMPC: minmax MPC implemented as a quadratic programming problem as in de la Peña *et al.* (2007). This controller minimizes the cost of the worst possible scenario of the tree at each time sample *k*.

In particular, 100 different simulations were carried out for the test. The disturbances used in the simulations were based on real rainfall data from The Netherlands, and their severity ranges over the whole spectrum of possible scenarios. Specifically, the maximum inflow into the drainage system of each simulation was chosen between 0 and 500 m^3/s, while the pumping capacity of the system under test was only 75 m^3/s. For each one of these simulations, the drainage system was controlled in a closed loop for 25 hours by each controller. The total cumulated cost of the 25 hours was computed according to the performance index defined in (3) and the parameters given in Table 1. The results can be seen in Table 4, and are consistent with the expectations about the performance of the controllers. Naturally, the PFMPC outperforms the rest of the controllers with a large difference, but, as has been stated previously, this controller is not realistic. Then, it can be seen how the TBMPC offers the best performance of the rest of the controllers. As was shown previously, the distributed versions of the TBMPC offer a very similar performance. The MPC controller with a probabilistic forecast is a step behind these controllers and shows a similar performance to the next one, the MMPC controller. Finally, the minmax offers the worst performance of all the controllers tested.

Table 4 | Total cumulated cost comparison

	Cumulated cost	
Controller	Mean	Std. dev.
PFMPC	1.42×10^7	4.77×10^7
TBMPC	1.56×10^7	4.87×10^7
DTBMPC1	1.57×10^7	4.86×10^7
DTBMPC3	1.56×10^7	4.87×10^7
DTBMPC5	1.58×10^7	4.86×10^7
DTBMPC7	1.57×10^7	4.86×10^7
DTBMPC9	1.58×10^7	4.87×10^7
ProbMPC	1.77×10^7	5.21×10^7
MMPC	1.77×10^7	5.21×10^7
MinmaxMPC	1.86×10^7	5.46×10^7

Remark. The results given in Table 4 show a high standard deviation due to the different severity of the stormy events that were used in the simulations. As can be seen in Table 1, the parameter with the highest impact in the performance index is the quadratic penalty on e_{zone}. For this reason, any simulation in which the water level is outside the allowed margin has a much higher cumulated cost in general.

CONCLUSIONS

In this paper, we have presented an implementation of TBMPC in a distributed fashion by means of dual decomposition. The distributed formulation allows us to apply TBMPC to problems with a higher number of optimization variables than traditional centralized model predictive controllers and sets the basis for distributed TBMPC. In addition, we have carried out experiments using a drainage water system as a benchmark in order to test how the number of iterations needed for convergence varies as different parameters change. It has been shown that a proper choice of the agreement horizon N_a and the disagreement threshold $\Delta u_{\max}$ can reduce the number of iterations in a significant manner, while keeping almost constant the TBMPC controller performance. Moreover, our results suggest that the relevance of the compatibility constraints is low from a control point of view. Nevertheless, the absence of these constraints makes the control signals calculated by the controller unrealistic for those time steps different from the first one. Finally, the TBMPC and its distributed versions have been compared with other controllers in the same benchmark. The results of our simulations showed that the TBMPC outperformed the rest of the alternatives considered, such as minmax MPC or MMPC, and it can be concluded that the TBMPC and its distributed versions are suitable controllers to deal with the uncertain inflows that are typical for this kind of open water system.

ACKNOWLEDGEMENTS

This research was supported by the BSIK project Next Generation Infrastructures (NGI), the Delft Research Center Next Generation Infrastructures, the European STREP project Hierarchical and distributed model predictive control (HD-MPC, contract number INFSO-ICT-223854) and the EU Network of Excellence Highly-complex and networked control systems (HYCON2, FP7/2007–2013 under grant agreement no. 257462). This work was carried out while J. M. Maestre was working for Delft University of Technology in The Delft Centre for Systems and Control.

REFERENCES

Ahn, H. 2000 Ground water drought management by a feed forward control method. *Journal of the American Water Resources Association* **36** (3), 501–510.

Boyd, S. & Vandenberghe, L. 2004 *Convex Optimization*. Cambridge University Press, Cambridge, UK.

Camacho, E. F. & Bordons, C. 2004 *Model Predictive Control in the Process Industry*, Second Edition. Springer-Verlag, London, UK.

Clemmens, A. J. & Schuurmans, J. 2004 Simple optimal downstream feedback canal controllers: theory. *Journal of Irrigation and Drainage Engineering* **130** (1), 26–34.

de la Peña, D., Alamo, T., Ramirez, D. & Camacho, E. 2007 Min-max model predictive control as a quadratic program. *IET Control Theory Applications* **1** (1), 328–333.

Escudero, L. F. 2000 Warsyp: a robust modeling approach for water resources system planning under uncertainty. *Annals of Operation Research* **95**, 313–339.

Giselsson, P. & Rantzer, A. 2010 Distributed model predictive control with suboptimality and stability guarantees. In: *2010 49th IEEE Conference on Decision and Control (CDC)*, Atlanta, Georgia, USA, pp. 7272–7277.

Gneiting, T. & Raftery, A. E. 2005 Weather forecasting with ensemble methods. *Science* **310** (5746), 248.

Gomez, M., Rodellar, J. & Mantecon, J. 2002 Predictive control method for decentralized operation of irrigation canals. *Applied Mathematical Modeling* **26** (11), 1039–1056.

Growe-Kuska, N., Heitsch, H. & Romisch, W. 2003 Scenario reduction and scenario tree construction for power management problems. In: *2003 IEEE Bologna Power Tech Conference Proceedings*, Bologna, Italy.

Maestre, J., Giselsson, P. & Rantzer, A. 2010 Distributed receding horizon kalman filter. In: *2010 49th IEEE Conference on Decision and Control (CDC)*, Atlanta, pp. 5068–5073.

Malaterre, P., Rogers, D. & Schuurmans, J. 1998 Classification of canal control algorithms. *Journal of Irrigation and Drainage Engineering* **124** (1), 3–10.

Muñoz de la Peña, D., Bemporad, A. & Alamo, T. 2005 Stochastic programming applied to model predictive control. In: *CDC-ECC '05, 44th IEEE Conference on Decision and Control 2005 and 2005 European Control Conference*, Seville, pp. 1361–1366.

Mulvey, J. M. & Ruszczynski, A. 1995 A new scenario decomposition method for large-scale stochastic optimization. *Operations Research* **43** (3), 477–490.

Negenborn, R. R., De Schutter, B. & Hellendoorn, H. 2006 Multi-agent model predictive control of transportation networks. In: *Proceedings of the 2006 IEEE International Conference on Networking, Sensing and Control (ICNSC 2006)*, Ft. Lauderdale, Florida, pp. 296–301.

Negenborn, R. R., De Schutter, B. & Hellendoorn, H. 2008 Multi-agent model predictive control for transportation networks: serial versus parallel schemes. *Engineering Applications of Artificial Intelligence* **21** (3), 353–366.

Negenborn, R. R., van Overloop, P. J., Keviczky, T. & De Schutter, B. 2009 Distributed model predictive control for irrigation canals. *Networks and Heterogeneous Media* **4** (2), 359–380.

Rantzer, A. 2009 Dynamic dual decomposition for distributed control. In: *American Control Conference, 2009*, ACC '09, St. Louis, MO, pp. 884–888.

Raso, L., van Overloop, P. J. & Schwanenberg, D. 2009 Decisions under uncertainty: use of flexible model predictive control on a drainage canal system. In: *Proceedings of the 9th Conference on Hydroinformatics*, Tianjin, China.

Raso, L., van de Giesen, N., Stive, P., Schwanenberg, D. & van Overloop, P. J. 2012 Tree structure generation from ensemble forecasts for real time control. *Hydrological Processes* **27** (1), 75–82.

Rockafellar, R. & Wets, R.-B. 1991 Scenario and policy aggregation in optimization under uncertainty. *Mathematics of Operation Research* **16** (1), 119–147.

Scattolini, R. 2009 Architectures for distributed and hierarchical model predictive control – a review. *Journal of Process Control* **19**, 723–731.

Shor, N. Z. 1985 *Minimization Methods for Nondifferentiable Functions*. Springer, Berlin, Germany.

Sutiene, K., Makackas, D. & Pranevicius, H. 2010 Multistage k-means clustering for scenario tree construction. *Informatica* **21** (1), 123–138.

van de Water, H. & Willems, J. 2002 The certainty equivalence property in stochastic control theory. *IEEE Transactions on Automatic Control* **26** (5), 1080–1087.

van Overloop, P. J. 2006 *Model Prevan de Giesendictive Control on Open Water Systems*. Ph.D. Thesis, Delft University of Technology, Delft, The Netherlands.

van Overloop, P. J., Weijs, S. & Dijkstra, S. 2008 Multiple model predictive control on a drainage canal system. *Control Engineering Practice* **16** (5), 531–540.

van Overloop, P. J., van Clemmens, A. J., Strand, R. J., Wagemaker, R. & Bautista, E. 2010 Real-time implementation of model predictive control on Maricopa-Stanfield irrigation and drainage district's WM canal. *Journal of Irrigation and Drainage Engineering* **136** (11), 747–756.

Weijs, S., van Leeuwen, E., van Overloop, P. & van de Giesen, N. 2007 Effect of uncertainties on the real-time operation of a lowland water system in The Netherlands. *International Association of Hydrological Sciences Publications* **313**, 463–470.

Witsenhausen, W. S. 1968 A minimax control problem for sampled linear systems. *IEEE Transactions on Automatic Control* **13** (1), 5–21.

Zafra-Cabeza, A., Maestre, J. M., Ridao, M. A., Camacho, E. F. & Sánchez, L. 2011 A hierarchical distributed model predictive control approach in irrigation canals: a risk mitigation perspective. *Journal of Process Control* **21** (5), 787–799.

First received 21 September 2011; accepted in revised form 2 July 2012. Available online 3 December 2012