Delft University of Technology

Delft Center for Systems and Control

Technical report 11-019

Fixed-profile load scheduling for large-scale irrigation channels*

Y. Li and B. De Schutter

Delft Center for Systems and Control
Delft University of Technology
Mekelweg 2, 2628 CD Delft
The Netherlands
phone: +31-15-278.24.73 (secretary)
URL: https://www.dcsc.tudelft.nl

Fixed-Profile Load Scheduling for Large-Scale Irrigation Channels[⋆]

Yuping Li[∗] Bart De Schutter[∗]

[∗] *Delft Center for Systems and Control, Delft University of Technology, Mekelweg 2, 2628 CD Delft, the Netherlands. (e-mail:{yuping.li,b.deschutter}@tudelft.nl)*

Abstract: The problem of fixed-profile load scheduling is considered for large-scale irrigation channels. Based on the analysis of the special structure of a channel under decentralised control, a predictive model is built on a pool-by-pool basis and a decomposition strategy of the scheduling problem is provided. The decomposition avoids excessive memory requirements in building the predictive model of the controlled plant and solving the formulated optimisation problem.

Keywords: Fixed-profile load scheduling, $\{0,1\}$ linear programming, large-scale systems, predictive model, constrained optimisation, hierarchical control.

1. INTRODUCTION

In large-scale irrigation networks, water is often distributed via open water channels under the power of gravity. The flow of water through the network is regulated by automated gates positioned along the channels (Cantoni et al., 2007; Mareels et al., 2005). The stretch of a channel between two gates is commonly called a pool. Water offtake points to farms and secondary channels are distributed along the pools. Typically they are at the downstream end of pools. As such, an important control objective is setpoint regulation of the water-levels immediately upstream of each gate, which enables flow demand at the (often gravity-powered) offtake points to be met without over-supplying. When the number of pools to be controlled is large and the gates widely dispersed, it is natural to employ a decentralised control structure. Fig. 1 shows a side view of a channel under decentralised feedback control. The flow into pool_i, denoted by u_i, equals to flow supplied by the upstream pool, v_{i-1}. Note u_i is actually the control action taken by controller K_i to regulate the water-level y_i to a relevant setpoint r_i, in the face of disturbances associated with variations of the uncontrolled offtake load d_i.
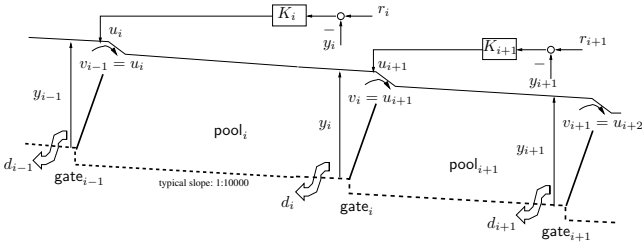

Fig. 1. Decentralised control of an open water channel

In practice, channel capacity is limited. This forces farmers to take water by placing orders. Moreover, the time-delay for water to travel from the upstream end to the downstream end of the pool limits the closed-loop bandwidth, which dampens the performance. Hence, the starting and ending of offtakes (d_i) induce transients (i.e. the water-level drops and rises from its setpoint). Such a transient response propagates to upstream pools as regulators take corrective actions (Cantoni et al., 2007; Clemmens and Schuurmans, 2004; Li and De Schutter, 2010). Indeed, water-levels stay at setpoints in steady state. Hence, the open water channel management objectives can be expressed in terms of constraints on the water-levels in each pool: upper bounds avoid water spillage over the banks of the channel; and lower bounds ensure a minimal channel capacity to supply water. In load scheduling, a set of offtakes (requested by farmers) is organised, which ensures that the water-level constraints are satisfied, in the face of transients associated with load changes. Moreover, from a farmer's perspective, a preferable solution would involve the smallest possible delay between the requested starting time and the time the load is scheduled. As a result, the scheduling can be expressed as an optimisation problem involving minimising the delay of water delivery subject to constraints.

The load scheduling sits on the higher-level of a two-level control hierarchy. On the lower-level, controllers are designed to ensure stability, robustness, good setpoint tracking, and disturbance rejection. The following load scheduling problem is considered in this paper, in particular, for large-scale irrigation channels. Given

- a linear controlled plant whose controller rejects disturbances associated with load variation,
- linear constraints on the transient response,
- load orders from users,

determine the smallest delay between the time the load is requested to start and the time it can be scheduled,

without violation of the constraints. [1] Note that preserving the profile of the requested load is a strong constraint on the scheduling task. Such a constraint corresponds to a specific production requirement, e.g. constant load over time in gravity-fed irrigation channels. This constraint actually comes from practice: since the offtake points are activated manually by users (to drop bars or gates), it is important to preserve the profile of a requested load (i.e. a constant flow over a period of time), in order for the farmers to activate the gate as less as possible (i.e. only for opening and closing it).

In (Alende et al., 2009), a predictive model of the controlled plant over a finite horizon is built as a function of the load to be scheduled. Then the scheduling problem is formulated as a combinatorial optimisation problem that can be rewritten as a $\{0, 1\}$ Linear Programming problem. However, when applied in load scheduling for large-scale irrigation channels, such a formulation has several limitations mainly due to computational issues with the size of the predictive model and the time to solve the constrained integer optimisation problem. In this paper, the special structure of an irrigation channel under decentralised control is studied. [2] It is shown that to overcome the associated computational issues, it is useful to build the predictive model on a pool-by-pool basis, with the interconnection between controlled pools as a constraint in the formulation of the scheduling problem. To decrease the computing time in solving the optimisation (scheduling) problem, a decomposition strategy is suggested, with which the number of decision variables and constraints decreases sharply.

The paper is organised as follows. Section 2 discusses the fixed-profile load scheduling problem for irrigation channels as formulated in (Alende et al., 2009). By analysing the special structure of a channel under decentralised control, a decomposition of this scheduling scheme is suggested in Section 3. In Section 4, simulation results are given to compare the proposed decomposition strategy with the centralised load scheduling scheme given in (Alende et al., 2009). A brief summary is finally given in Section 5.

2. FIXED-PROFILE LOAD SCHEDULING PROBLEM FOR A CONTROLLED IRRIGATION CHANNEL

The formulation of the fixed-profile load scheduling problem is discussed in (Alende et al., 2009). The idea is to predict the behaviour of the controlled channel (composed of N pools) over a finite horizon as a function of the delays between the requested offtakes and the scheduled ones. Throughout the prediction horizon, the water-levels are constrained. Given a cost function penalising the overall delays, the resulting formulation is a Nonlinear Integer

---

[1] It is important to differentiate: – the (transportation) time-delay water takes to travel from the upstream to the downstream end of a pool, and – the (delivery) delay between the time an offtake is requested to start and the time it is scheduled. The first delay is a known characteristic of the system, the second one is the decision variable of the scheduling problem.

[2] In fact, the decomposition strategy proposed in this paper can also be applied to load scheduling for channels under distributed control (Cantoni et al., 2007; Li and Cantoni, 2008).
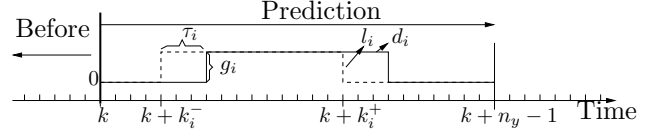

Fig. 2. A requested load and one possible schedule

Programming problem. Further, identifying that the number of values the delays can take in a finite horizon is finite, by applying a change of variables, the problem is formulated as a $\{0, 1\}$ Linear Programming problem as follows.

Let l_i, see the dashed line in Fig. 2, denote the requested offtake from $pool_i$. Within the prediction horizon (i.e. from the time slot k to $k+n_y-1$), the profile of l_i is represented by $l_i^{[k,k+n_y-1]} = \left[\mathbf{0}^{k_i^-}, g_i \times \mathbf{1}^{k_i^+ - k_i^-}, \mathbf{0}^{n_y-k_i^+} \right]^T$, where $\mathbf{0}^n$ represents a vector of n 0's and $\mathbf{1}^n$ a vector of n 1's, $k+k_i^-$ and $k+k_i^+$ denote the starting and stopping time of the requested offtake, respectively, and g_i is the magnitude of the offtake. One possible schedule of the requested offtake, d_i, is shown in the figure (see the solid line). In particular,

$$d_i^{[k,k+n_y-1]} = J^{\tau_i} l_i^{[k,k+n_y-1]},$$

where $\tau_i \in \{0, 1, 2, \ldots\}$ is the delivery delay between the starting time of the requested offtake l_i and the starting time of the scheduled offtake d_i; [3] and J is an $n_y \times n_y$ lower shift matrix,

$$J = \begin{bmatrix} 0 & & \cdots & & 0 \\ 1 & 0 & & & \vdots \\ & \ddots & \ddots & & \\ 0 & & & 1 & 0 \end{bmatrix}.$$

Note that l_i has n_i $(= n_y - k_i^+ - T_i^{\max})$ possible schedules, where $T_i^{\max}$ is the maximal duration of the transients in $pool_1$ to $pool_i$ caused by stopping of the offtake d_i. [4] Such a setting of n_i ensures all the response transients to be entirely contained inside the prediction horizon. Let $M_i \in \mathbb{R}^{n_y \times n_i}$ represent all possible delayed versions of the requested load, any schedule of l_i can be represented by

$$d_i = \left[J^0 l_i | J^1 l_i | \ldots | J^{n_i-1} l_i \right] z_i =: M_i z_i, \quad (1)$$

where z_i is a vector of size n_i, with only one element equal to 1 and the others 0, which corresponds to only one schedule being selected. For a string of N pools, the offtake-load scheduling problem is then formulated as

$$\min_{z_i} \sum_{i=1}^{N} h_i^T z_i \quad (2)$$

s.t.

$$\hat{y}^{[k+1,k+n_y]} = \Gamma x(k) + \Omega r^{[k,k+n_y-1]} + \Psi \Pi (Mz); \quad (3)$$

$$\underline{y}^{[k+1,k+n_y]} \leq \hat{y}^{[k+1,k+n_y]} \leq \overline{y}^{[k+1,k+n_y]}; \quad (4)$$

$$\sum_{m=1}^{n_i} (z_i)_m = 1, \quad (5)$$

$$z_i \in \{0, 1\}^{n_i} \text{ for } i = 1, \ldots, N. \quad (6)$$

In the cost function (2), h_i^T is a weight penalising the delays. For example, one can set $h_i^T = [1, 2, \ldots, n_i]$ for

---

[3] In this paper, scheduling an offtake in advance of its requested time, i.e. $\tau_i < 0$, is not considered from the practical perspective.

[4] This considers the transient propagating to upstream pools under distant-downstream control as shown in Fig. 1.

all i and hence no offtake has a particular priority. The equality constraint (3) is a process model of the controlled channel, it predicts the water-levels of the N pools as a linear function of the decision variables z_i, where

$$Mz := [M_1 z_1, \ldots, M_N z_N]^T, \ \hat{y}^{[k+1,k+n_y]} := \begin{bmatrix} \hat{y}(k+1) \\ \vdots \\ \hat{y}(k+n_y) \end{bmatrix}$$

with $\hat{y}(k) := \begin{bmatrix} \hat{y}_1(k) \\ \vdots \\ \hat{y}_N(k) \end{bmatrix}$, where $\hat{y}_i(k)$ is the estimate of the

water-level y_i at time step k, $r(k) := \begin{bmatrix} r_1(k) \\ \vdots \\ r_N(k) \end{bmatrix}$ with $r_i(k)$

the setpoint of water-level y_i at k, and Γ, Ω, Ψ, and Π are linear matrices, as defined in Section 3.1. Constraint (6) requires that the n_i elements in z_i be either 0 or 1. Together with the constraint (5), it is required that only one component of the variable z_i is 1, which represents the requested load can only be scheduled once.

Remark 1. If one relaxes the constraint (6) as

$$0 \le (z_i)_m \le 1 \text{ for } m = 1, \ldots, n_i \text{ and } i = 1, \ldots, N,$$

the original optimisation problem is reduced to a load-scheduling problem when the profile of the loads are not constrained. In particular, the parametrisation is such that the scheduled load profile is composed of fractions of the requested one with different delays. Note the relaxed optimisation problem is a linear programming problem. However, as discussed in Section 1, based on the practical operations in feeding water to farms, we focus on fixed-profile load scheduling in this paper.

The formulation (2-6) is a mixed-integer linear program, for which efficient algorithms exist (Floudas, 1998). However, as pointed out in (Alende et al., 2009), it has several limitations mainly due to computational issues:

- The size of the predictive model is proportional to the number of pools and the length of the prediction horizon.
- The computation time when solving the constrained integer optimisation problem increases drastically with the number of decision variable and constraints.

The facts that for offtake-load scheduling the forecast horizon n_y is often large (e.g. in the simulation in (Alende et al., 2009) $n_y = 480$, to forecast a schedule for 80 hours under a practical consideration) and that the number of pools in a channel could be above 30 make the previous scheduling strategy impractical for large-scale irrigation networks. To overcome this, a decomposition strategy is suggested in the next section, which is based on the analysis of the special structure of a string of pools under decentralised control.

3. DECOMPOSITION OF THE LOAD SCHEDULING PROBLEM

For an irrigation channel under decentralised distant-downstream control, the information exchange between subsystems is one-by-one from downstream to upstream (i.e. $v_i = u_{i+1}$ as shown in Fig. 1). Such a special structure makes it feasible to schedule load requests from downstream to upstream in sequence, i.e. first schedule for d_N, then d_{N-1}, ..., at last for d_1. Hence the schedul-

ing optimisation problem introduced in Section 2 can be decomposed. Moreover, in such a decomposition, the interconnections between subsystems are represented as a function of the already scheduled load. We can then build a predictive model based on each controlled pool, which reduces the scheduling problem further in such a way that when scheduling for d_i, the water-level constraints of the upstream pools (i.e. $pool_1$ to $pool_{i-1}$) can be transferred to new water-level constraints of $pool_i$.

3.1 Structure analysis of the controlled plant

For a string of pools under decentralised control (see Fig. 1), each subsystem (i.e. each controlled pool) can have a discrete-time state-space representation as

$$x_i(k+1) = A_i x_i(k) + B_{r_i} r_i(k) + B_{d_i} d_i(k) \\ + A_{p_i} x_{i+1}(k) \quad (7)$$
$$y_i(k) = C_i x_i(k). \quad (8)$$

Note the interconnection between pools (i.e. flow out of $pool_i$ equals to flow into $pool_{i+1}$), $v_i = u_{i+1}$, is represented by the term $A_{p_i} x_{i+1}$. A system with N subsystems can then have a model as

$$x(k+1) = \bar{A} x(k) + \bar{B}_r r(k) + \bar{B}_d d(k) \quad (9)$$
$$y(k) = \bar{C} x(k) \quad (10)$$

where $x(k) := \begin{bmatrix} x_1(k) \\ \vdots \\ x_N(k) \end{bmatrix}$, $y(k) := \begin{bmatrix} y_1(k) \\ \vdots \\ y_N(k) \end{bmatrix}$, $d(k) :=$

$\begin{bmatrix} d_1(k) \\ \vdots \\ d_N(k) \end{bmatrix}$, $\bar{A} = \begin{bmatrix} A_1 & A_{p_1} & & \\ & A_2 & A_{p_2} & \\ & & \ddots & \ddots \\ & & & A_N \end{bmatrix}$, $\bar{B}_r = \mathrm{diag}\,(B_{r_1}, \ldots, B_{r_N})$,

$\bar{B}_d = \mathrm{diag}\,(B_{d_1}, \ldots, B_{d_N})$, and $\bar{C} = \mathrm{diag}\,(C_1, \ldots, C_N)$.

As already indicated in Section 2, the essence of determining the scheduling scheme is to optimise, over the feasible delivery delay between the scheduled load and the request load, forecasts of transient behaviour. The process model is the essential element of the scheduler. By writing the dynamic equation of the discrete model (7-8) recursively as discussed in (Maciejowski, 2002), the following process model is obtained that makes a prediction $\hat{y}$ of the water levels y over a finite horizon of n_y slots of length T_s:[5]

$$\hat{y}^{[k+1,k+n_y]} = \Gamma x(k) + \Omega r^{[k,k+n_y-1]} + \Psi \tilde{d}^{[k,k+n_y-1]} \\ + \Psi d^{[k,k+n_y-1]}, \quad (11)$$

where Γ is a column matrix with the j-th row as $\bar{C} \bar{A}^j$, Ω and Ψ are lower-triangular, Toeplitz matrices with the j-th row given by $(\bar{C} \bar{A}^{j-1} \bar{B}_r, \ldots, \bar{C} \bar{B}_r, 0, \ldots, 0)$ and $(\bar{C} \bar{A}^{j-1} \bar{B}_d, \ldots, \bar{C} \bar{B}_d, 0, \ldots, 0)$ respectively. In (11), the load disturbance is split into two parts – d is the scheduling vector of the new load and $\tilde{d}$ is the vector of the already scheduled load, which is assumed to be known over the prediction horizon n_y.[6]

It is observed that the interconnection between the subsystems is only represented in the off-diagonal entries in $\bar{A}$, and that $\bar{B}_r$, $\bar{B}_d$ and $\bar{C}$ have diagonal structure. As a

---

[5] The discrete time model (7-8) is transformed from a continuous-time model (see Section 4.1) with sampling time T_s.

[6] It is assumed that a scheduled offtake will be executed as it is planned, hence the already scheduled loads influence the scheduling result of the newly requested loads.

result, the non-zero entries in Γ, Ω and Ψ, i.e. $\bar{C}\bar{A}^j$, $\bar{C}\bar{A}^j\bar{B}_r$ and $\bar{C}\bar{A}^j\bar{B}_d$ respectively (for $j = 1, \cdots, n_y$), have upper-triangular structure. Next, we change the stackings of input variables and output variables. Then the predictive model has the following alternative expression:

$$
\begin{pmatrix} \hat{y}_1^{[k+1,k+n_y]} \\ \vdots \\ \hat{y}_N^{[k+1,k+n_y]} \end{pmatrix} = \Pi\Gamma x(k) + \Pi\Omega\Pi \begin{pmatrix} r_1^{[k,k+n_y-1]} \\ \vdots \\ r_N^{[k,k+n_y-1]} \end{pmatrix}
$$
$$
+ \Pi\Psi\Pi \begin{pmatrix} \tilde{d}_1^{[k,k+n_y-1]} \\ \vdots \\ \tilde{d}_N^{[k,k+n_y-1]} \end{pmatrix} + \Pi\Psi\Pi \begin{pmatrix} d_1^{[k,k+n_y-1]} \\ \vdots \\ d_N^{[k,k+n_y-1]} \end{pmatrix},
$$

where $\Pi \in \{0,1\}^{(n_y \times N) \times (n_y \times N)}$ is a mapping, stacking the variables in an appropriate way. Note that $\Pi\Omega\Pi$ and $\Pi\Psi\Pi$ have a form of

$$
\begin{bmatrix}
* & & & * & & \\
\vdots & \ddots & \cdots & \vdots & \ddots & \\
* & \cdots & * & * & \cdots & * \\
& & & \ddots & & \vdots \\
& & & & * & \\
& & & \vdots & & \ddots \\
& & & * & \cdots & *
\end{bmatrix}
$$

with $*$ non-zero entries. Such a block upper-diagonal structure is the result of the special interconnection structure of a string of pools under decentralised control. It implies that the following scheduling strategy can be applied:

(1) $i \leftarrow N$;
(2) Plan d_i – Set $d_1, \ldots, d_{i-1}$ equal to $\mathbf{0}^{n_y}$;[7] find the optimal $d_i(k) = l_i(k - \tau_i)$ such that $\underline{y}_h^{[k+1,k+n_y]} \leq \hat{y}_h^{[k+1,k+n_y]} \leq \overline{y}_h^{[k+1,k+n_y]}$ for $h = 1, \cdots, i$.[8] If no feasible d_i found, set $d_i = \mathbf{0}^{n_y}$.
(3) Augment $\tilde{d}_i$ with the already scheduled d_i: $\tilde{d}_i \leftarrow \tilde{d}_i + d_i$;
(4) $i \leftarrow i - 1$; if $i = 0$, stop; else, go to (2).

The scheduling sequence from d_N to d_1 is a natural selection– it takes into account the fact that with distant downstream control structure, the water-level error caused by water offtakes in one pool propagates to all the upstream pools; hence any schedule of offtake load d_i will influence scheduling of $d_1, \ldots, d_{i-1}$. Note that such a decomposition gives priority to offtake load requests in the downstream pools.

3.2 Building the predictive model on a pool-by-pool basis

Note that in the step (2) of the load scheduling strategy proposed above, when scheduling for d_i, one needs to check the upper and lower bounds of water-levels in pool$_1$ to pool$_i$. When the number of pools is large, the computing time to solve such a decomposed (combinatorial) optimisation problem is still an issue. The following discussions shows that one can go further in decomposing the scheduling problem such that the scheduling of d_i is based on some newly set upper and lower bounds of water-levels in pool$_i$. The new bounds involve the nominal constraints of y_1 to y_{i-1} on the scheduling of d_i. The idea is to build the predictive model on a pool-by-pool basis.

---

[7] $\mathbf{0}^{n_y}$ represents a vector composed of n_y 0's.
[8] As introduced in Section 1, optimal means τ_i minimal.

Each controlled pool has the following state-space representation:

$$
\begin{bmatrix} x_i(k+1) \\ y_i(k) \end{bmatrix} = \begin{bmatrix} A_i & B_{r_i} & B_{d_i} & A_{p_i} \\ C_i & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_i(k) \\ r_i(k) \\ d_i(k) \\ v_i(k) \end{bmatrix}, \qquad (12)
$$

and as discussed in Section 4.1, each decentralised feedback controller K_i in Fig. 1 can be represented as:

$$
\begin{bmatrix} x_i^K(k+1) \\ u_i(k) \end{bmatrix} = \begin{bmatrix} A_i^K & B_{e_i} \\ C_i^K & 0 \end{bmatrix} \begin{bmatrix} x_i^K(k) \\ r_i(k) - y_i(k) \end{bmatrix}. \qquad (13)
$$

Again, note that the interconnection between the controlled pools $v_i = u_{i+1}$. The predictions of the response of a controlled pool over a finite horizon of n_y slots (of duration T_s) can be computed as follows: for $i = 1, \ldots, N$

$$
\hat{y}_i^{[k+1,k+n_y]} = \Gamma_i x_i(k) + \Omega_i r_i^{[k,k+n_y-1]} + \Psi_i \tilde{d}_i^{[k,k+n_y-1]}
$$
$$
+ \Psi_i d_i^{[k,k+n_y-1]} + \Upsilon_i v_i^{[k,k+n_y-1]} \qquad (14)
$$
$$
v_i^{[k+1,k+n_y]} = u_{i+1}^{[k+1,k+n_y]} \qquad (15)
$$
$$
u_i^{[k+1,k+n_y]} = \Gamma_i^K x_i^K(k) + \Omega_i^K r_i^{[k,k+n_y-1]} - \Omega_i^K \hat{y}_i^{[k,k+n_y-1]} \qquad (16)
$$

where Γ_i and Γ_i^K are column matrices with respectively $C_i A_i^j$ and $C_i^K (A_i^K)^j$ as the j-th row (for $j = 1, \ldots, n_y$), and Ω_i, Ψ_i, Υ_i and Ω_i^K are lower-triangular, Toeplitz matrices with the j-th row given by

$$
\left(C_i A_i^{j-1} B_{r_i}, \ldots, C_i B_{r_i}, 0, \ldots, 0 \right),
$$
$$
\left(C_i A_i^{j-1} B_{d_i}, \ldots, C_i B_{d_i}, 0, \ldots, 0 \right),
$$
$$
\left(C_i A_i^{j-1} A_{p_i}, \ldots, C_i A_{p_i}, 0, \ldots, 0 \right),
$$

and $\left(C_i^K (A_i^K)^{j-1} B_{e_i}, \ldots, C_i^K B_{e_i}, 0, \ldots, 0 \right)$ respectively. As previously introduced, $\tilde{d}_i$ represents the already scheduled load in pool$_i$.

We schedule for d_i: Consider two neighbouring subsystems, subsystem i and subsystem $i-1$. Following the scheduling strategy suggested in Section 3.1, assume no new load requests in subsystem $i-1$, i.e. $d_{i-1}^{[k,k+n_y-1]} = \mathbf{0}^{n_y}$. Note that the influence of the scheduling of d_i on $\hat{y}_{i-1}$ (i.e. $\hat{y}_{i-1}^{[k+1,k+n_y]} = f(d_i^{[k,k+n_y-1]})$ with $f(\cdot)$ linear) is through v_{i-1}. From (14-16), the bounds that

$$
\underline{y}_{i-1}^{[k+1,k+n_y]} \leq f(d_i^{[k,k+n_y-1]}) \leq \overline{y}_{i-1}^{[k+1,k+n_y]}
$$

can be transformed to bounds on y_i:

$$
\underline{y}_{i,\text{cal}}^{[k+1,k+n_y]} = \left(\Upsilon_{i-1} \Omega_i^K \right)^{-1} \left(\tilde{y}_{i-1}^{[k+1,k+n_y]} - \overline{y}_{i-1}^{[k+1,k+n_y]} \right)
$$
$$
+ \left(\Omega_i^K \right)^{-1} \Gamma_i^K x_i^K(k) + r_i^{[k,k+n_y-1]} \qquad (17)
$$
$$
\overline{y}_{i,\text{cal}}^{[k+1,k+n_y]} = \left(\Upsilon_{i-1} \Omega_i^K \right)^{-1} \left(\tilde{y}_{i-1}^{[k+1,k+n_y]} - \underline{y}_{i-1}^{[k+1,k+n_y]} \right)
$$
$$
+ \left(\Omega_i^K \right)^{-1} \Gamma_i^K x_i^K(k) + r_i^{[k,k+n_y-1]} \qquad (18)
$$

where $\tilde{y}_i^{[k+1,k+n_y]} := \Gamma_i x_i(k) + \Omega_i r_i^{[k,k+n_y-1]} + \Psi_i \tilde{d}_i^{[k,k+n_y-1]}$.

Remark 2. In (17) and (18) it is assumed that the lower-triangular, Toeplitz matrices Υ_{i-1} and Ω_i^K are non-singular. This can be guaranteed if the diagonal entries are non-zero, which is the case for the pools with decentralised control (see Section 4.1). Note the inverses are also lower triangular, Toeplitz matrices.

Correspondingly, the new bounds on y_i are the intersections of $\left[\underline{y}_{i,\text{cal}}, \overline{y}_{i,\text{cal}}\right]$ and $\left[\underline{y}_i, \overline{y}_i,\right]$: for $j = k+1, \ldots, k+n_y$

$$\underline{y}_{i,\text{new}}(j) = \max\left\{\underline{y}_{i,\text{cal}}(j), \underline{y}_i(j)\right\} \tag{19}$$

$$\overline{y}_{i,\text{new}}(j) = \min\left\{\overline{y}_{i,\text{cal}}(j), \overline{y}_i(j)\right\} \tag{20}$$

In this way, one can set new water-level bounds for load scheduling, in a sequence from upstream to downstream. Hence, we have the following two-stage load scheduling strategy for large-scale irrigation networks:

(1) Calculate new water-level bounds for scheduling, from upstream to downstream:
  (a) $i \leftarrow 1$, set $\underline{y}_{1,\text{new}}^{[k+1,k+n_y]} \leftarrow \underline{y}_1^{[k+1,k+n_y]}$ and $\overline{y}_{1,\text{new}}^{[k+1,k+n_y]} \leftarrow \overline{y}_1^{[k+1,k+n_y]}$.
  (b) $i \leftarrow i + 1$, set $\underline{y}_{i-1}^{[k+1,k+n_y]} \leftarrow \underline{y}_{i-1,\text{new}}^{[k+1,k+n_y]}$ and $\overline{y}_{i-1}^{[k+1,k+n_y]} \leftarrow \overline{y}_{i-1,\text{new}}^{[k+1,k+n_y]}$. Calculate $\underline{y}_{i,\text{new}}$ by (17), (19) and $\overline{y}_{i,\text{new}}$ by (18), (20).
  (c) If $i < N$, go to (1b); else go to (2).
(2) Schedule offtake loads from downstream to upstream:
  (a) $i \leftarrow N$; $v_N^{[k,k+n_y-1]} \leftarrow \mathbf{0}^{n_y}$.[9]
  (b) Represent $\hat{y}_i$ as a function of z_i by replacing d_i in (14) with $M_i z_i$ as in (1).
  (c) Find the optimal z_i by solving
  $$\min_{z_i} h_i^T z_i$$
  s.t.
  $$\underline{y}_{i,\text{new}}^{[k+1,k+n_y]} \le \hat{y}^{[k+1,k+n_y]} \le \overline{y}_{i,\text{new}}^{[k+1,k+n_y]};$$
  $$\sum_{m=1}^{n_i} (z_i)_m = 1,$$
  $$z_i \in \{0,1\}^{n_i}.$$
  Calculate d_i from (1). If no feasible solution found, $d_i^{[k,k+n_y-1]} \leftarrow \mathbf{0}^{n_y}$.
  (d) Calculate $\hat{y}_i^{[k+1,k+n_y]}$ by taking $d_i^{[k,k+n_y-1]}$ into (14).
  (e) $i \leftarrow i - 1$. If $i > 0$, calculate $v_i^{[k+1,k+n_y]}$ by (15-16), go to (2b); else, end.

3.3 Discussion on the computational complexity

The load scheduling solution based on the decomposition strategy investigated in Section 3.2 is suboptimal compared to the optimal solution, obtained by the centralised scheduling scheme introduced in Section 2. However, when the number of pools is large, the computational complexity of the decomposed scheduling strategy, which considers the special structure of the controlled plant, is light compared to the centralised scheduling scheme. This is significant for management of large-scale irrigation networks. For example, to schedule N offtake loads from N pools (i.e. one offtake per pool), each requested load, l_i, has n_i possible schedules. Hence the number of decision variables for each demand is n_i. With the centralised scheduling scheme, the number of decision variables is $\sum_{i=1}^{N} n_i$; the number of constraints is $\sum_{i=1}^{N} n_i + N \times 2n_y + N$; and there are

$\prod_{i=1}^{N} n_i$ combinations of possible schedules.[10] While with the decomposition scheduling strategy, when one schedules for d_i, the number of decision variables is n_i; the number of constraints is $n_i + 2n_y + 1$; the number of the combinations of all possible schedules is n_i.

In fact, for large-scale irrigation networks, the memory space gained by building the model on a pool-by-pool basis (see Section 3.2) is substantial compared with that on a channel basis, i.e. all pools at once (see Section 2 and 3.1). In particular, the transfer function matrix from d_i to y_i is represented by a block Ψ_i (see (14)); while the impact on y_i by interaction between controlled pools, i.e. v_i, is represented by a block Υ_i, which has a similar structure as Ψ_i. In total, to represent the mapping from $d := [d_1, \ldots, d_N]^T$ to $y := [y_1, \ldots, y_N]^T$ for N controlled pools, $2N - 1$ such blocks are requested. In contrast, in the predictive model constructed on a channel basis, such a relationship is represented by a block-triangular matrix Ψ (see (11)), which consists of $\frac{N(N+1)}{2}$ such (nonzero) blocks, each with the same size as Ψ_i.

4. CASE STUDIES

We study two cases in this section. First, the decomposed scheduling strategy is applied in two pools of East Goulburn Main (EGM) Channel, Victoria, Australia. The scheduling result, compared with that from the centralised scheduling strategy, is suboptimal. Then, to compare the computational complexity, both the scheduling strategies are tested in a string of ten pools of EGM.

4.1 Predictive model building for a controlled pool

Following Cantoni et al. (2007), each controlled pool in a channel as shown in Fig. 1 has the following continuous state-space realisation:

$$\dot{x}_i(t) = \tilde{A}_i x_i(t) + \tilde{B}_{r_i} r_i(t) + \tilde{B}_{d_i} d_i(t) + \tilde{A}_{p_i} v_i(t)$$
$$y_i(t) = \tilde{C}_i x_i(t)$$

with $\tilde{A}_i = \begin{bmatrix} 0 & c_{\text{in},i} & -c_{\text{in},i} & 0 \\ 0 & \frac{-2}{t_{\text{d},i}} & \frac{4}{t_{\text{d},i}} & 0 \\ \frac{-\kappa_i}{\rho_i} & 0 & 0 & 1 \\ \frac{-\kappa_i(\rho_i-\phi_i)}{\phi_i\rho_i^2} & 0 & 0 & \frac{-1}{\rho_i} \end{bmatrix}$, $\tilde{A}_{p_i} = \begin{bmatrix} -c_{\text{out},i} \\ 0 \\ 0 \\ 0 \end{bmatrix}$,

$\tilde{B}_{d_i} = \begin{bmatrix} -c_{\text{out},i} \\ 0 \\ 0 \\ 0 \end{bmatrix}$, $\tilde{B}_{r_i} = \begin{bmatrix} 0 \\ 0 \\ \frac{\kappa_i}{\rho_i} \\ \frac{\kappa_i(\rho_i-\phi_i)}{\phi_i\rho_i^2} \end{bmatrix}$, $\tilde{C}_i = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}$, where

$c_{\text{in},i}$ and $c_{\text{out},i}$ are discharge coefficients, functions of the pool surface area and the gate width; and $t_{\text{d},i}$ is the internal time-delay that the water takes to travel from the upstream end to the downstream end of a pool;[11] κ_i, ρ_i and ϕ_i are parameters of the decentralised feedback controller K_i, which is a PI compensator with a low-pass filter. Note that the interconnection between neighbouring (controlled) pools can be expressed as $v_i = u_{i+1}$. The

---

[9] Under distant-downstream control, setting a boundary condition of $v_N = 0$ is indeed possible (Cantoni et al., 2007).

[10] Although there exist efficient algorithms to solve the formulated mixed-integer linear program, when the number of pools is large, the computation load can still be heavy such that the solving engine is out of memory, see Case 2 in Section 4.2.

[11] A first-order Padé approximation is used to represent the transportation time-delay $t_{\text{d},i}$.

Pool	$c_{\text{in},i}$	$c_{\text{out},i}$	τ_i
Campbells	0.055	0.036	5 min
Schifferlies	0.017	0.026	6 min

Controller	κ_i	ϕ_i	ρ_i
1	0.74	71.83	8.52
2	1.19	141.27	16.75

Table 1. Parameters of (controlled) pools

feedback controller K_i has the following continuous state-space representation:

$$\dot{x}_i^K(t) = \tilde{A}_i^K x_i^K(t) + \tilde{B}_{e_i}(r_i(t) - y_i(t))$$
$$u_i(t) = \tilde{C}_i^K x_i^K(t)$$

where $\tilde{A}_i^K = \begin{bmatrix} 0 & \frac{\kappa_i(\rho_i - \phi_i)}{\phi_i \rho_i^2} \\ 0 & \frac{-1}{\rho_i} \end{bmatrix}$, $\tilde{B}_{e_i} = \begin{bmatrix} \frac{\kappa_i}{\rho_i} \\ 1 \end{bmatrix}$ and $\tilde{C}_i^K = \begin{bmatrix} 1 & 0 \end{bmatrix}$.

To build the predictive model, a discrete-time state-space model (12-13) is employed. This can be obtained by directly converting the continuous model through a zero-order hold. The sampling interval T_s should be of duration small enough to capture the whole relevant dynamics of the system. In the case studies in Section 4.2, the sampling time is set to 10 minutes.

4.2 Simulation results

The scheduling procedure introduced in Section 3.2 is applied in the following case studies.

Case 1 First, schedule offtake requests from the last two pools (i.e. Campbells and Schifferlies) of EGM. The parameters of controlled pools are given in Table 1. In this case, 3 requested offtakes in each pool are considered. The notation $d_{i,j}$ represents the j−th offtake happening in pool$_i$. Since all load-disturbances happening in the same pool are identically modelled, it is relevant to adapt the notation used in the optimisation formulation in step (2b) as follows: the overall load of a pool is noted $d_i = \sum_{j=1}^3 d_{i,j}(t)$; $M_{i,j}$ and $z_{i,j}$ are respectively the selection of the eligible solution and the $\{0,1\}$ decision variable associated to the requested offtake $l_{i,j}$. The cost function is chosen as $\min_{z_{i,j}} \sum_{j=1}^3 [1, 2, \ldots, n_{i,j}] z_{i,j}$, which minimises the overall delivery delay in a pool.

The prediction horizon $n_y = 480$ (of 10 minutes), which corresponds to a forecast of 80 hours. The possible scheduled delays for each offtake is restricted to multiples of one hour (i.e. 6 ($\times$10min)). In the simulation, the system is at steady-state at the beginning of the horizon. Given the already scheduled offtakes ($\tilde{d}_1^{[k,k+n_y-1]}$ and $\tilde{d}_2^{[k,k+n_y-1]}$ shown as in the top plots for pool$_1$ and pool$_2$ (Fig. 3) respectively), the task is to schedule three requested offtakes per pool (represented by the dash-dotted line in the figure) such that the constraints on the water-level of each pool are satisfied. The requested flow is in the range of $20 - 35$ Ml/day, which is reasonable considering the pool characteristics. The lower bound and the upper bound on the water-levels are fixed, at 9.4 m and 9.7 m for pool$_1$ and 9.5 m and 9.7 m for pool$_2$, throughout the simulation period. The setpoint r_i changes from 9.5 m to 9.6 m at 3600 min for pool$_1$ and from 9.56 m to 9.62 m at 1200 min for pool$_2$.
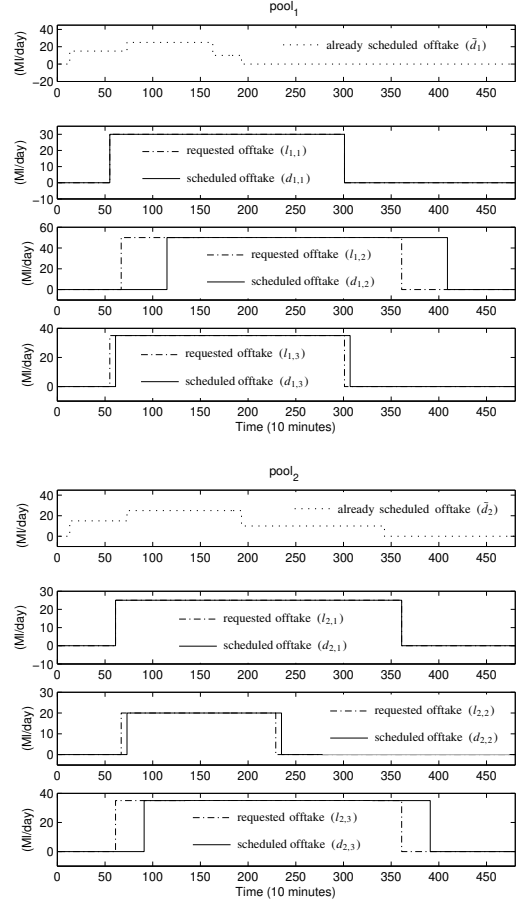

Fig. 3. Scheduling of requested offtakes (with the decomposed load scheduling strategy)

Centralised load scheduling (hours)		Decomposed load scheduling (hours)	
$\tau_{1,1} = 1$	$\tau_{2,1} = 4$	$\tau_{1,1} = 0$	$\tau_{2,1} = 0$
$\tau_{1,2} = 3$	$\tau_{2,2} = 0$	$\tau_{1,2} = 8$	$\tau_{2,2} = 1$
$\tau_{1,3} = 0$	$\tau_{2,3} = 6$	$\tau_{1,3} = 1$	$\tau_{2,3} = 5$
$\sum_{j=1}^3 \tau_{1,j} = 4$	$\sum_{j=1}^3 \tau_{2,j} = 10$	$\sum_{j=1}^3 \tau_{1,j} = 9$	$\sum_{j=1}^3 \tau_{2,j} = 6$
$\sum_{i=1}^2 \sum_{j=1}^3 \tau_{i,j} = 14$		$\sum_{i=1}^2 \sum_{j=1}^3 \tau_{i,j} = 15$	

Table 2. Comparison of scheduling results

Forecasting of the influence of the *requested* offtake loads on the system dynamic is represented by the dash-dotted lines in Fig. 4. The upper bound and lower bound (constraints) on the water-levels ($\underline{y}_i$ and $\bar{y}_i$) are violated at some time instants (around 750 min and around 3750 min) in the prediction horizon. In contrast, under the scheduled offtakes, the dynamics of the system is within the water-level constraints (see solid lines in Fig. 4). The scheduling results are shown by the solid line in Fig. 3. For comparison, the same simulation scenario is set to test the centralised load scheduling in (Alende et al., 2009) (see Section 2). The results are shown in Table 2. We see that compared to the centralised load scheduling (from which the total time-delay is 14 hours), the decomposed load scheduling provides a sub-optimal solution (i.e. the
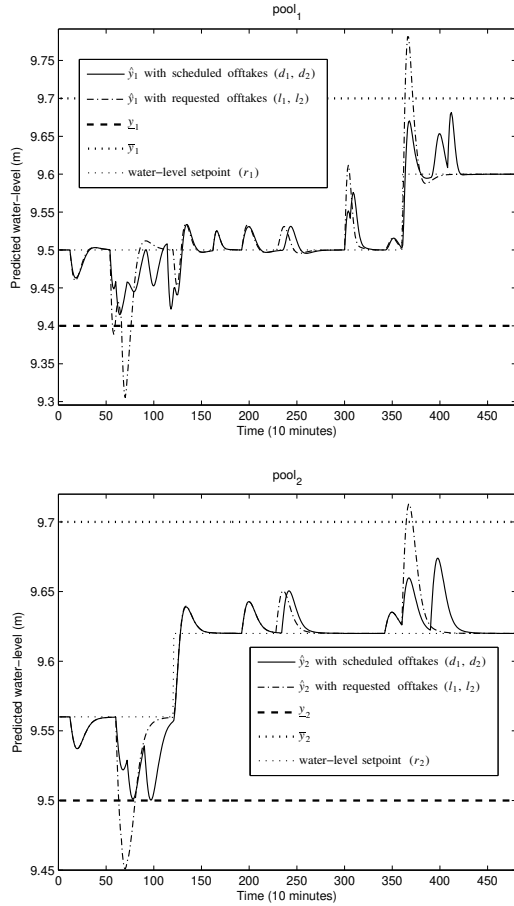
Fig. 4. Forecasting of water-levels (with the decomposed load scheduling strategy)

Centralised load scheduling	
number of constraints	$\sum_{i=1}^{10} \sum_{j=1}^{3} n_{i,j} + N\,(2n_y + 1) = 11160$
number of possible scheduling combinations	$\prod_{i=1}^{10} \prod_{j=1}^{3} n_{i,j} = 6.7044 \times 10^{21}$
Decomposed load scheduling (averagely, for each pool)	
number of constraints	1116
number of possible scheduling combinations	155

Table 3. Comparison of computational complexity

with the decomposed scheduling strategy, the computation load decreases drastically.

5. CONCLUSIONS

The problem of load scheduling for large-scale irrigation channels is considered. Based on the analysis of the special structure of open water channels under decentralised control, a decomposition of the scheduling problem is discussed. The solution could be suboptimal compared to the optimal solution to the scheduling problem initially formulated in Alende et al. (2009). However, such a decomposition scheme avoids computational issues, including memory requirements and computing time, which is significant for a large-scale system.

REFERENCES

Alende, J., Li, Y., and Cantoni, M. (2009). A {0,1} linear program for fixed-profile load scheduling and demand management in automated irrigation channels. In *Proceedings of the 48th IEEE Conference on Decision and Control*, Shanghai, China, pages 597–602.

Cantoni, M., Weyer, E., Li, Y., Ooi, S.K., Mareels, I., and Ryan, M.(2007) Control of large-scale irrigation networks. In *Special Issue of the Proceedings of the IEEE on the Emerging Technology of Networked Control Systems*, volume 95(1), pages 75–91.

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

Floudas, C.A. (1998). *Nonlinear and Mixed-Integer Optimization*. Oxford University Press.

Li, Y. and Cantoni, M. (2008). Distributed controller design for open water channels. In *Proceedings of the 17th IFAC World Congress*, Seoul, South Korea, pages 10033–10038.

Li, Y. and De Schutter, B. (2010). Control of a string of identical pools using non-identical feedback controllers. To be presented in *Proceedings of the 49th IEEE Conference on Decision and Control*, Atlanta, GA, USA.

Maciejowski, J.M. (2002). *Predictive Control with Constraints*. Prentice Hall, England.

Mareels, I., Weyer, E., Ooi, S.K., Cantoni, M., Li, Y., and Nair, G. (2005). Systems engineering for irrigation systems: Successes and challenges. In *Annual Reviews in Control*, volume 29(2), pages 191–204.

total time delay is 15 hours). As discussed in Section 3.1, the decomposed load scheduling strategy gives priority to offtake requests in the downstream pools, i.e. $\tau_2 = 6$ hours instead of 10 hours (from the centralised load scheduling).

Case 2 To see how much the decomposed scheduling strategy gain in decreasing computational complexity, we then check the case of load scheduling for a channel of 10 (controlled) pools. When the centralised scheduling scheme is applied, the construction of the predictive model on a channel basis (i.e. all pools at once) takes 184 seconds (on a Core2 CPU 3GHz, with 4GB of RAM). In contrast, when the decomposed scheduling strategy is applied, the construction of the predictive model on a pool-by-pool basis takes the CPU time of 15 seconds (actually 1.5 seconds for each controlled pool). To solve the constrained mixed-integer optimisation problem (e.g. we schedule for 3 offtakes per pool), applying the centralised scheduling scheme, the solving capacity is exceeded. In contrast, when the decomposed scheduling strategy is used, the total computing time for solving the constrained mixed-integer optimisation problem is 17 seconds (1.7 seconds in average for offtakes in each pool). Table 3 compares the computation complexity for such a case. Although efficient algorithms exist (e.g. in the simulation, Branch and Bound method is used) to solve the optimisation problem, the computational complexity could still be a problem using the centralised scheduling scheme. However,