Partitioning Approach oriented to the Decentralised Predictive Control of Large-Scale Systems

C. Ocampo-Martinez*, S. Bovo, V. Puig

Institut de Robòtica i Informàtica Industrial (CSIC-UPC)
Llorens i Artigas, 4-6, 2nd floor
08028 Barcelona, Spain

Abstract

In this paper, a partitioning approach for large-scale systems based on graph-theory is presented. The algorithm starts with the translation of the system model into a graph representation. Once the system graph is obtained, the problem of graph partitioning is then solved. The resultant partition consists in a set of non-overlapping subgraphs whose number of vertices is as similar as possible and the number of interconnecting edges between them is minimal. To achieve this goal, the proposed algorithm applies a set of procedures based on identifying the highly-connected subgraphs with balanced number of internal and external connections. In order to illustrate the use and application of the proposed partitioning approach, it is used to decompose a dynamical model of the Barcelona drinking water network (DWN). Moreover, a hierarchical-like DMPC strategy is designed and applied over the resultant set of partitions in order to assess the closed-loop performance. Results obtained when used several simulation scenarios show the effectiveness of both the partitioning approach and the DMPC strategy in terms of the reduced computational burden and, at the same time, of the admissible loss of performance in contrast to a centralised MPC strategy.

Keywords: graph partitioning algorithms, decentralised MPC, large-scale networked systems, drinking water networks

1. Introduction

Large-scale systems (LSS) present control theory with new challenges due to the large size of the plant and of its model [1, 2]. The goal to be achieved with control methods for this kind of systems is to obtain a reasonable solution with a reasonable effort in modelling, designing and implementing the controller.

Model-based Predictive Control (MPC) has been proved to be one of the advanced control techniques widely accepted for the control of LSS [3]. Applications to different large-scale infrastructures as drinking water networks [4], sewer networks [5], open-flow channel networks [6] or electrical networks [7] proves the applicability of this technique. The main reason is due to once obtained the plant dynamical model, the MPC design just consists in expressing the desired performance specifications through different control objectives (e.g., weights on tracking errors and actuator efforts as in classical linear quadratic regulation), and constraints on system variables (e.g., minima/maxima of selected process variables and/or their rates of change) which are necessary to ensure process safety and asset health. The rest of the MPC design is automatic: the given model, constraints, and weights define an optimal control problem over a finite time horizon in the future (for this reason the approach is said predictive). This is translated into an equivalent optimisation problem and solved on line to obtain an optimal sequence of future control moves. Only the first of these moves is applied to the process, as at the next time step a new optimal control problem is solved, to exploit the information coming from fresh new measurements. In this way, an open-loop design methodology (i.e., optimal control) is transformed into a feedback one.

Nevertheless, the main hurdle for MPC control (as any other control technique) when applied to LSS in a centralised way, is the non-scalability. The reason is that a huge control model is needed, being difficult to maintain/update and which needs to be rebuilt on every change of the system configuration, e.g., when some part of the system should be stopped because of maintenance actions or malfunctions. Subsequently, a model change would require re-tuning the centralised controller. It is obvious that the cost of setting up and maintaining the monolithic solution of the control problem is prohibitive. A way of circumventing these issues might be by looking into *decentralised* MPC (DMPC) or *distributed* MPC techniques, where networked local MPC controllers are in charge of controlling part of the entire system. The main difference between distributed and decentralized MPC is that the former *uses* negotiations and re-computations of local control actions within the sampling period to increase the level of cooperation, whereas the latter does not (at the benefit of computation time, but at the cost of optimality).

The industrial success of the traditional centralised MPC (CMPC) drives now a new interest in this old area of distributed control, and distributed MPC has become one of the hottest topics in process control in the early 21st century, worldwide.

*Corresponding author. Tel: +34 93 401 5752; Fax: +34 93 401 5750.
Email address: cocampo@iri.upc.edu (C. Ocampo-Martinez)

Thus, two research projects (HDMPC [8] and WIDE [9]) are currently being carried out in Europe, both focused on the development of decentralised and distributed MPC techniques. Few works have been recently published in this area; see, e.g., [10, 11, 12, 13, 14, 15], among others.

However, in order to apply decentralised or distributed MPC approaches to LSS, there is a prior problem to be solved: the system decomposition into subsystems. The importance of this issue has already been noticed in classic control books addressing the decentralised control of LSS as [1, 2]. The decomposition of the system in subsystems could be carried out during the modelling of the process by identifying subsystems as parts of the system on the basis of physical insight, intuition or experience. But, when a large-scale complex system with many states, inputs and outputs is considered, it may be difficult, even impossible, to obtain partitions by physical reasoning. A more appealing alternative is to develop systematic methods, which can be used to decompose a given system by extracting information from its structure and representing it as a graph. Then, this structural information can be analysed by using methods coming from graph theory. Consequently, the problem of system decomposition into subsystems leads to the problem of graph partitioning, i.e., the decomposition of graph into subgraphs.

Graph partitioning is an important problem with extensive application in scientific computing [16], optimisation, VLSI design [17], task partitioning for parallel processing, control of cascading failures, among others. Several algorithms coping this problem exist in the literature as presented in a brief review in Section 2. However, the development of graph partitioning algorithms that allow the decomposition of LSS into subsystems for being used in decentralised or distributed MPC is still very incipient and available methods are quite limited. In [2], a hierarchical LBT decomposition that leads to a input-reachable hierarchy for some particular systems is presented. A more general approach is based on the ε-decomposition method, which is based on decomposing the system in weakly coupled subsystems (see also [2]). The algorithm proceeds sequentially disconnecting the edges of the system graph that are smaller than a prescribed threshold ε and identifying the disconnected subgraph of the resulting graph. The obtained subsystems correspond to the subsystems with mutual coupling smaller or equal than ε. However, the tuning of this parameter is not a trivial issue and only a trial and error approach is currently available.

The main contribution of this paper is to go one step further in the development of subsystem decomposition methods for LSS by proposing a new automatic decomposition algorithm based on graph partitioning. The aim of the proposed method is to provide a decomposition consisting of a set of non-overlapping subgraphs whose number of vertices is as similar as possible and the number of interconnecting edges between them is minimal. To achieve this goal, the proposed algorithm is composed of a set of graph-theory-based procedures, which identify the highly-connected subgraphs with balanced number of internal and external connections. A real case study based on the Barcelona DWN is used to test the proposed subsystem decomposition methodology using a recently proposed DMPC scheme [18].

The paper is structured as follows: in Section 2, the dynamical system decomposition into subsystems seen as a graph partitioning problem is stated. Section 3 presents the proposed partitioning approach for dynamical systems. Section 4 describes the case study considered in the paper. Section 5 discusses both the application of the proposed graph partitioning approach, and the implementation of a hierarchical-like DMPC strategy over the case study, and presents the most relevant results. Finally, conclusions and directions for further work are reported in Section 6.

Notation

In the sequel, let $\mathbb{R}$ and $\mathbb{Z}$ denote the set of real numbers and the set of integer numbers, respectively. Moreover, $\mathbb{Z}_{\geq c} \triangleq \{k \in \mathbb{Z} \ : \ k \geq c\}$, for some $c \in \mathbb{Z}$, $\mathbb{R}_+ \triangleq \mathbb{R} \setminus (-\infty, 0)$ is the set of non-negative real numbers, and $\#V$ denotes the cardinality of subset V. The set difference of two sets A and B is defined as $A - B = \{x : x \in A \wedge x \notin B\}$.

2. Problem Formulation using Graph Theory

A graph can be defined as an abstract representation of a set of objects from a certain collection, where some pairs of objects are connected by links. The interconnected elements are typically called *vertices* while the connection links are called *edges*. These latter elements may be *directed* (asymmetric) or *undirected* (symmetric) according to their connection features, what makes that the whole graph is directed or undirected as well. It is also possible to distinguish graphs whether or not their vertices and edges are weighted (weighted/unweighted graphs).

Consider a dynamical system represented in general form by the state-space equations

$$x^+ = g(x, u, d), \tag{1a}$$

$$y = h(x, u, d), \tag{1b}$$

where $x \in \mathbb{R}^n$ and $x^+ \in \mathbb{R}^n$ are, respectively, the current and successor system states, $u \in \mathbb{R}^m$ is the system input and $d \in \mathbb{R}^p$ is a bounded process disturbance. Moreover, $g : \mathbb{R}^n \mapsto \mathbb{R}$ is the states mapping function and $h : \mathbb{R}^m \mapsto \mathbb{R}$ corresponds with the output mapping function. Suppose now that it is desired to decompose (1) into subsystems. With this aim, the graph representation of the system model (1) is determined (by using the system topology) and incidence matrix I_M is then stated, which describes the connections (edges) between the graph vertices (system inputs, outputs and states). Without loss of generality, I_M and the directonality of the edges are derived from the relation between system equations (rows of I_M) and system variables (columns of I_M), as proposed by [2, 19, 20]. There are alternative matrix representations for a (directed) graph such as the *adjacency matrix* and the *Laplacian matrix* (see [21]), which are related to the matrix representation used in this paper. Once I_M has been obtained from the system directed graph (digraph), the problem of the decomposition into subsystems can be formulated in terms of partitioning the corresponding graph into subgraphs. Since such partitioning is oriented to the application of a decentralised control strategy (in particular, DMPC),

the resultant subgraphs should have the following features (see [1, 2]):

- nearly the same number of vertices;
- few connections between the subgraphs.

These features guarantee that the obtained subgraphs have a similar size, fact that balances computations between subsystem controllers and allows minimising communications between them. Hence, the problem of graph partitioning can be more formally established as follows:

Problem 1 (Standard Graph Partitioning). *Given a graph $G(V, E)$, where V denotes the set of vertices, E is the set of edges, and $k \in \mathbb{Z}_{\geq 1}$, find k subsets $V_1, V_2, \ldots, V_k$ of V such that*

1. $\bigcup_{i=1}^{k} V_i = V$,
2. $V_i \cap V_j = \emptyset$, *for* $i \in \{1, 2, \ldots, k\}$, $j \in \{1, 2, \ldots, k\}$, $i \neq j$,
3. $\#V_1 \approx \#V_2 \approx \cdots \approx \#V_k$,
4. *the* cut size, *i.e., the number of edges with endpoints in different subsets V_i, is minimised.*

Remark 2.1. *Defining the vertex-based weight of a subset V_i as*

$$\Omega_i \triangleq \sum_{j=1}^{\#V_i} \omega_i^j, \tag{2}$$

where ω_i^j corresponds to the weight of the j-th vertex of the subset V_i, the following condition should be added to Problem 1 in the case of weighted graph partitioning:

- $\Omega_i \approx \Omega/k$, *with* $i \in \{1, 2, \ldots, k\}$, *where*

$$\Omega \triangleq \sum_{i=1}^{k} \Omega_i. \tag{3}$$

Remark 2.2. *Conditions 3 and 4 of Problem 1 are of high interest from the decentralised control point of view since they are related to the degree of interconexion between resultant subsystems and their size balance, respectively.*

Graph partitioning is considered as a $\mathcal{NP}$-complete problem [2]. However, it can be solved in polynomial time for $\#V_i = 2$ (Kernighan-Lin algorithm) [22, 23]. Since this condition is very restrictive for large-scale graphs, alternatives for graph partitioning based on fundamented heuristics are properly accepted. Two main classes of successful heuristics have evolved over the years, trying to achieve the proper trade off between partitioning speed and quality. They are the *minimum-degree-based* ordering algorithms (MDB), and the *graph-partitioning-based* ordering algorithms (GPB) [24].

The MDB algorithms are local greedy heuristic, which reorder the columns of a symmetric sparse matrix such that the column with the fewest non-zero elements at a given iteration of factorisation was eliminated at the next iteration [25, 19]. GPB algorithms regard to the symmetric sparse matrix as the adjacency matrix of a graph and follow a *divide-and-conquer* strategy to label the vertices of the graph by partitioning it into smaller subgraphs [26].

The initial success of MDB algorithms prompted intense research to improve their run and quality (multiple minimum degree and approximate minimum degree). However, later works suggest that the GPB algorithms are capable of producing better-quality ordering than the MDB algorithms for finite-element problems, while staying within a small constant factor of the run time of MDB algorithms [27, 28].

3. Partitioning Approach for Dynamical Systems

As said in the Introduction, the main contribution of this paper consists in proposing a partitioning algorithm, as much automatised as possible, through which a partition of a dynamical system can be found, which allows its decomposition in subsystems. This algorithm requires to represent the dynamical system as a graph, which can obtained from the system structure [2].

3.1. Main Algorithm

The partitioning algorithm proposed in this paper follows some ideas developed in [24] for graph partitioning purposes (mainly the strategy based on GPB algorithms mentioned in Section 2). However, some refining steps have been added as well as some of the original procedures have been drastically changed in order to find partitions oriented to split dynamical networked systems. Hence, the different parts/routines of the main proposed algorithm are presented and explained in sections below. The current version of the algorithm is though to be used off-line, i.e., the partitioning of the system is not carried out on-line. A further improvement could be to adapt the proposed algorithm such that the partitioning could be done on-line when some structural change of the network occurs. In this way, the potential benefit of using a DMPC approach described in the Introduction could be fully exploited.

3.1.1. Start up

This procedure requires the definition of the graph, i.e., *the incidence matrix*[1] I_M, which describes the connections between the graph vertices, their directionality and, in some cases, the weight of each edge.

[1]The *incidence matrix* of a directed graph $G(V, E)$, denoted as I_M, is defined such that

$$I_{Mij} = \begin{cases} -1 & \text{if the edge } x_j \text{ leaves vertex } v_i, \\ 1 & \text{if the edge } x_j \text{ enters vertex } v_i, \\ 0 & \text{otherwise.} \end{cases}$$

This matrix has dimensions $\varphi \times \eta_e$, where φ corresponds with the total number of vertices and η_e denotes de total number of edges [21]. Additionally, the weight of the j-th vertex, denoted as ω^j, for $j = 1, 2, \ldots, \varphi$, where $\varphi \triangleq \#V$, is computed. The weight ω^j represents the number of edges connected to this vertex. Moreover, ω^j is also known as the *vertex degree* [29].

3.1.2. Preliminary partitioning

This procedure performs a preliminary automatic partitioning of the graph as follows. The vertex $v_j \in V$, for $j \in \{1, 2, \ldots, \varphi\}$, with maximum weight ω is found and defined as the centre of the first subgraph G_1. Then, all vertices connected to this vertex of maximum weight are assigned to G_1. At this point, the set of non-selected vertices is defined as

$$V_r \triangleq \{v_j \in V \,:\, v_j \notin V_1\}.$$

This procedure is now repeated for all vertices $v_j \in V_r$ (now for $j = \{1, 2, \ldots, \#V_r\}$) until V_r is empty, after the corresponding updating. This routine highlights the subgraphs of higher connectivity. The resultant subgraphs with just one vertex are merged to the closest subgraph. Once a set of subgraphs $G_i(V_i, E_i)$, for $i = 1, 2, \ldots, k$, is obtained, it is possible to determine some useful indexes for the entire graph and each one of the resu subgraphs. These indexes are:

- $\varphi_i \triangleq \#V_i$ (from now on called *subgraph internal weight* of G_i);
- ε_i, denoted as the *cut size*[2] of the subgraph G_i (from now on called *subgraph external weight* of G_i);
- $\varphi_{\max} \triangleq \max_i \varphi_i$, for $i = 1, 2, \ldots, k$;
- $\bar{\varphi} \triangleq \frac{1}{k} \sum_{i=1}^{k} \varphi_i$ (arithmetic mean).

Notice that at this stage, the number k of subgraphs is obtained in an automatic way so it is not imposed.

Remark 3.1. *Notice that introducing the set $\tilde{E}_a \subset E$, defined as the set of edges with endpoints in other subgraphs different to G_a, the representation of subgraphs G_i such that*

$$\bigcup_{i=1}^{k} G_i = G,$$

can be slightly modified to $G_i(V_i, E_i, \tilde{E}_i)$ for completeness purposes. Also notice that $\varepsilon_i \triangleq \#\tilde{E}_i$.

3.1.3. Uncoarsening - Internal balance

This procedure aims at the reduction of the number of subgraphs, trying to achieve similar internal weights for all of them. This process starts determining the set

$$M = \{G_i, i = 1, 2, \ldots, m \,:\, \varphi_i \leq \bar{\varphi}\}. \tag{4}$$

For each $G_i \in M$, the set of neighbour[3] subgraphs, denoted as N_i, is determined and expressed as

$$N_i = \{G_j,\ j = 1, 2, \ldots, h_i \,:\, G_j \text{ is neighbour of } G_i\}, \tag{5}$$

with $h_i = \#N_i$. If the condition

$$\varphi_i + \varphi_j \leq \bar{\varphi}, \quad i \in \{1, 2, \ldots, m\},\ j \in \{1, 2, \ldots, h_i\} \tag{6}$$

holds for $G_i \in M$ and $G_j \in N_i$, then these two subgraphs are merged. If there are two or more subgraphs $G_j \in N_i$ such that (6) holds, the subgraph $G_j \in N_i$ with minimum internal weight is selected. Once two subgraphs are merged, $\bar{\varphi}$ is updated.

This procedure is iterated until no additional merging was possible. It is considered that the internal balance has been achieved when either

- $\bar{\varphi} \leq \varphi_i \leq \varphi_{\max}$, for $i = 1, 2, \ldots, k$, or
- G_i with $\varphi_i \leq \bar{\varphi}$ cannot be merged with any of its neighbours since the φ associated to the resultant subgraph might be greater than $\varphi_{\max}$.

3.1.4. Refining - External balance

This procedure aims at the reduction of the cut size of the resultant subgraphs. To achieve this goal, define ω_i^j as the degree of the j-th vertex of the i-th subgraph, with $j \in \{1, 2, \ldots, \varphi_i\}$ and $i \in \{1, 2, \ldots, k\}$. From this definition, two indexes can be stated:

- the *vertex internal degree*, denoted as $\hat{\omega}_i^j$, which represents the number of connections of the vertex $v_j \in V_i$, for $j \in \{1, 2, \ldots, \varphi_i\}$, $i \in \{1, 2, \ldots, k\}$, with other vertices $v_p \in V_i$, $p \in \{1, 2, \ldots, \varphi_i\}$, $p \neq j$;
- the *vertex external degree*, denoted as $\check{\omega}_i^j$, which represents the number of connections of the vertex $v_j \in V_i$, for $j \in \{1, 2, \ldots, \varphi_i\}$, $i \in \{1, 2, \ldots, k\}$, with other vertices $v_p \in V_q$, $p \in \{1, 2, \ldots, \varphi_q\}$, $q \in \{1, 2, \ldots, k\}$, $q \neq i$.

Hence, for a given vertex $v_j \in V_i$, if $\hat{\omega}_i^j < \check{\omega}_i^j$, then vertex v_j is moved from subgraph $G_i(V_i, E_i, \tilde{E}_i)$ to the subgraph in which most of its edges have their endpoint (like in the AVL tree algorithm [29]). All indexes should be updated for the k subgraphs and the next vertex is analysed. This procedure will last until each subgraph vertex fulfils $\hat{\omega}_i^j \geq \check{\omega}_i^j$.

3.1.5. The Complete Algorithm

Algorithm 1 collects all the procedures/routines mentioned and explained before. Hence, applying this algorithm to the graph associated to a given dynamical system, the expected result consists of a set of subgraphs which determines a particular system decomposition. This set P is then defined as

$$P = \left\{ G_i,\ i = 1, 2, \ldots, k \,:\, \bigcup_{i=1}^{k} G_i = G \right\}. \tag{7}$$

3.2. Auxiliary Routines

Despite Algorithm 1 yields an automatic partitioning of a given graph, it does not imply that the resultant set P follows the pre-established requirements stated in Problem 1. In this sense,

[2] See Problem 1.

[3] Two subgraphs are called *neighbours* if they are contiguous and share edges (see, e.g., [30] among many others).

Algorithm 1 Graph partitioning algorithm

$I_M \leftarrow$ System topology
% Start up
$G(V, E) \leftarrow I_M$
for $j = 1$ to φ **do**
  Compute ω^j
end for
% Preliminary partitioning
$V_r \leftarrow V$, $i = 1$
repeat
  Find $v \in V_r$ with maximum ω
  $V_i \leftarrow v$ and all its neighbour vertices
  $V_r \triangleq V - \left\{ \bigcup_{h=1}^{i} V_h \right\}$
  $i = i + 1$
until $V_r = \emptyset$
for $i = 1$ to k **do** *% Compute some indexes*
  $\varphi_i \triangleq \#V_i$ *% internal weight*
  $\varepsilon_i \triangleq \#\tilde{E}_i$ *% external weight*
end for
$\varphi_{\max} \triangleq \max_i \varphi_i$
$\bar{\varphi} \triangleq \frac{1}{k} \sum_{i=1}^{k} \varphi_i$ *% arithmetic mean*
% Uncoarsening
Compute M *% see* (4)
$b_{\text{int}} =$ false *% Internal balance*
while $b_{\text{int}} =$ false **do**
  for $i = 1$ to m **do**
    Compute N_i *% see* (5)
    for $j = 1$ to h **do**
      if $\varphi_i + \varphi_j \leq \bar{\varphi}$ **then** *% see* (6)
        $G_* = G_i \cup G_j$
        $G_{\text{new}} \leftarrow G_*$ with minimum φ_*
        Update $\bar{\varphi}$
      end if
    end for
  end for
  Update φ_i
  % Refining
  $b_{\text{ext}} =$ false *% External balance*
  while $b_{\text{ext}} =$ false **do**
    for $i = 1$ to k **do**
      for $j = 1$ to φ_i **do**
        Compute $\hat{\omega}_i^j$ and $\check{\omega}_i^j$
        if $\hat{\omega}_i^j < \check{\omega}_i^j$ **then**
          Move v^j from G_i to its neighbour
        end if
        Update φ_i, $\bar{\varphi}$, $\varphi_{\max}$
      end for
    end for
    Update all indexes
    Check external balance (nodes)
  end while
  Check internal balance (subgraphs)
end while
return P *% see* (7)

complementary routines can be useful for improving the partitioning process according to the considered application. Additional auxiliary routines could be added such that the generated partitioning takes into account the control performance that would be achieved when used in decentralised or distributed MPC control.

3.2.1. Pre-filtering

In general, the resultant solution given by the Algorithm 1 is nearly appropriate in terms of $\hat{\omega}$ and $\check{\omega}$, but it highly depends on the topology and complexity of the graph. For this reason, in order to obtain a better graph partitioning, sometimes it can be useful to make a *Pre-filtering* routine, where all the vertexes with $\omega = 1$ are virtually merged to this vertex that shares its unique edge. This procedure creates *supranodes*, which should be properly recognised at the moment of determining the partitioning of the dynamical system from the decomposition of its associated graph. Moreover, doing the manual merging of those vertices reduces the work done by subsequent routines.

3.2.2. Post-filtering

On the other hand, suppose that after partitioning a given graph $G(V, E)$ by using Algorithm 1, all the k resultant subgraphs fulfil

$$\bar{\varphi} \leq \varphi_i \leq \varphi_{\max}, \quad \text{for} \quad i \in \{1, 2, \ldots, k\}. \tag{8}$$

However, the following situation could occur. Suppose a subgraph G_a with $\varphi_a \ll \bar{\varphi}$, which is placed next to a subgraph G_b and fulfils (8). The merging of subgraphs G_a and G_b, expressed as $G_c \triangleq G_a \cup G_b$, is not allowed since $\varphi_c \geq \varphi_{\max}$. The *Post-filtering* routine implements an approximation and a parametrisation, i.e., by adding a small tolerance δ, the existence of the resultant subgraph G_c is now allowed since $\varphi_c \leq \varphi_{\max} + \delta$. This relaxation allows to have less subgraphs but with higher complexity and internal weight.

3.2.3. Anti-oscillation

This procedure leads to solve a possible issue when the *refining (external balance)* routine is run. When a vertex is moved from one subgraph to another according to its internal and external degrees, there exists the possibility of doing this movement during an infinite time if there is no specification of routine ending. Therefore, the refining routine is then run within a for loop and the parameter ρ is set as the maximum number of iterations that this procedure is executed. Afterwards, since the resulting set of subgraphs is stored at each iteration $t' \in \mathbb{Z}$, $t' = \{1, 2, \ldots, \rho\}$, the configuration of k subgraphs with minor ε_i, for $i = 1, 2, \ldots, k$, can be chosen.

3.3. Some Practical Issues

Given that the partitioning algorithm proposed in this paper is mainly thought for performing decentralised control of LSS, several features could be taken into account to achieve a convenient system partitioning and less complex controller designs. For instance, an additional routine that would restrict the connection of subgraphs with unidirectional edges would be very useful since a pure hierarchical control scheme can be straightforwardly implemented, decreasing the inherent loss of performance of a decentralised control scheme.

4. Case Study Description

The Barcelona drinking water transport network (DWN) has been used as the case study to illustrate the performance of the proposed partitioning approach and the subsequent employment of the DMPC strategy reported in [18].

4.1. System Description

The Barcelona DWN, managed by Aguas de Barcelona, S.A. (AGBAR), not only supplies drinking water to Barcelona city but also to the metropolitan area. The sources of water are the Ter and Llobregat rivers, which are regulated at their head by some dams with an overall capacity of 600 cubic hectometres. Currently, there are four drinking water treatment plants (WTP): the Abrera and Sant Joan Despí plants, which extract water from the Llobregat river, the Cardedeu plant, which extracts water from Ter river, and the Besòs plant, which treats the underground flows from the aquifer of the Besòs river. There are also several underground sources (wells) that can provide water through pumping stations. Those different water sources currently provide a flow of around 7 m^3/s. The water flow from each source is limited, what implies different water prices depending on water treatments and legal extraction canons.

The Barcelona DWN is structurally organised in two layers[4]. The upper layer, named as *transport network*, links the water treatment plants with the reservoirs distributed all over the city. The lower layer, named *distribution network* is sectorised in subnetworks. Each subnetwork links a reservoir with each consumer. This paper is focused on the transport network. Thus, each subnetwork of the distribution network is modelled as a demand sector. The demand of each sector is characterised by a demand pattern, which can be predicted by using a time-series model [31]. The control system of the transport network is also organised in two layers (see Figure 1). The upper layer is in charge of the global control of the network, establishing the set-points of the regulatory controllers at the lower layer. Regulatory controllers are of PID type, while the supervisory layer controller is of MPC type. Regulatory controllers hide the network non-linear behaviour to the supervisory controller. This allows the MPC supervisory controller to use a flow-based control-oriented linear model.

4.2. Control-oriented Modelling

Control-oriented modelling principles for DWNs have been widely presented in the literature, see [4, 32]. In order to obtain a control-oriented model of the DWN, the constitutive network elements as well as their basic relationships should be discussed. The reader is referred to the aforementioned references for further details of DWN modelling and specific insights related to the case study of this paper.

Consider the main physical constraints of the DWN given by the variables related to the tank volumes and manipulated flows.

[4]The proposed decomposition between transportation and distribution part is only possible if the hydraulic couplings are weak as in the case of Barcelona DWN. In other water networks, the strong hydraulic coupling could prevent from the application of such a decomposition.

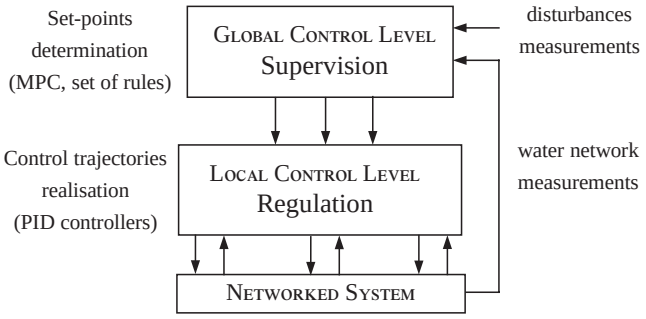

Figure 1: Hierarchical structure for RTC system

For the case of tank volumes, the physical constraint related to the range of volume capacities for the i-th tank is expressed as

$$x_i^{\min} \leq x_i(t) \leq x_i^{\max}, \quad \forall\, t, \tag{9}$$

where $x_i^{\min}$ and $x_i^{\max}$ denote the minimum and maximum volume capacity, respectively, given in m^3 and t denotes the discrete time. Moreover, $x_i^{\min} \geq 0$. On the other hand, the physical constraints related to manipulated flows through the system actuators are expressed as

$$u_i^{\min} \leq u_i(t) \leq u_i^{\max}, \quad \forall\, t, \tag{10}$$

where $u_i^{\min}$ and $u_i^{\max}$ denote the minimum and the maximum flow capacity, respectively, given in m^3/s. Moreover, $u_i^{\min} \geq 0$.

By considering the mass balance in tanks, the control-oriented model of the DWN in discrete-time state-space form can be written as

$$x(t+1) = A\,x(t) + B\,u(t) + B_p\,d(t), \tag{11}$$

where $x \in \mathbb{X} \subseteq \mathbb{R}_+^n$ is the state vector corresponding to the water volumes of the n tanks, $u \in \mathbb{U} \subseteq \mathbb{R}_+^m$ represents the vector of manipulated flows through the m actuators (pumps and valves), and $d \in \mathbb{D} \subseteq \mathbb{R}_+^p$ corresponds to the vector of the p water demands (sectors of consume). A, B, and B_p are system matrices of suitable dimensions. Since the demands can be forecasted, they are assumed to be known. Thus, d is a known vector of non-negative elements, containing the measured disturbances affecting the system. By also including static relations at network nodes, model (11) can be further rewritten as

$$x(t+1) = A\,x(t) + \Gamma\,\upsilon(t), \tag{12a}$$

$$E_1\,\upsilon(t) = E_2, \tag{12b}$$

where $\Gamma = [B \quad B_p]$, $\upsilon(t) = [u(t)^T \quad d(t)^T]^T$, and E_1, E_2 are matrices of suitable dimensions dictated by the network topology.

The Barcelona DWN model (12) contains a total amount of 67 tanks and 121 actuators, these latter divided in 46 pumps and 75 valves. Moreover, the network has 88 demand sectors and 16 water nodes. Both the demand episodes and the network calibration/simulation set-up are provided by AGBAR. Figure 3 (further below) depicts the considered network.

4.3. System Management Criteria

As said before, AGBAR provides the management policies for the Barcelona DWN given their knowledge of the system and the performance objectives that it is to be reached commonly in this kind of networked systems. Thus, these criteria are described as follows.

4.3.1. Minimising water production and transport costs

The main economic costs associated with drinking water production (treatment) are due to chemicals, legal canons, and electricity costs. Delivering drinking water with appropriate pressure levels through the water transport network involves important electricity costs in pumping stations. The corresponding performance figure to be minimised is expressed as

$$f_1(t) = W_e\ (\alpha_1 + \alpha_2(t))\, u(t), \tag{13}$$

where α_1 corresponds to a known vector related to the economic costs of the water according to the selected source (treatment plant, dwell, etc.) and $\alpha_2(t)$ is associated with the economic cost of the flow through certain actuators (pumps only) and their control cost (pumping). Note the time variance of α_2 due to the fact that pumping effort prices have different values according to the time of the day (electricity costs). The weight W_e is the penalty associated with economic costs with respect to the other objectives that will be included in the MPC optimisation problem. Also notice the linear nature of expression (13) is given by the unidirectional feature of all the manipulated flows.

4.3.2. Safety storage term

The satisfaction of water demands should be fulfilled at every time instant. However, some risk prevention mechanisms should be introduced in the tank management so that the stored volume is preferably maintained around a given safety value in case of emergency, and to guarantee future water availability in case of inaccurate demand forecasts. A quadratic expression for this concept is used and written as follows:

$$f_2(t) = (x(t) - \beta\, x^{\max})^T\, W_x\, (x(t) - \beta\, x^{\max}), \tag{14}$$

where β is a term which determines the safety volume to be considered for the control law computation and matrix W_x defines the weight of the objective in the cost function. This term prevents the controller from keeping the lowest possible water volumes in the tanks, which would reduce robustness to demand forecast inaccuracy.

4.3.3. Smoothness of the control actions

Pumping stations should avoid excessive switching: valves should operate smoothly in order to avoid harmful transients in the pressurised pipes, which can in turn lead to poor pipe conditions. Similarly, water flows requested from treatment plants must have a smooth profile due to plant operational constraints. Notice that the considered control-oriented modelling does not take into account pressure dynamics, hence a lower-level controller that keeps the desired flow is assumed. The use of a smooth reference surely *helps* the performance of such low-level controller. To smooth out the control action of MPC, the following third term is included in the objective function in order to penalise variations $\Delta u(t) = u(t) - u(t-1)$ of the control signal between consecutive sampling intervals

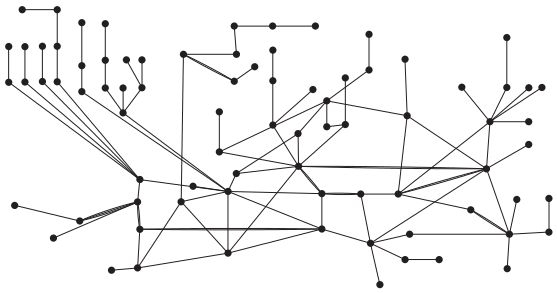

Figure 2: Graph related to the Barcelona DWN model after the application of the Pre-filtering routine

$$f_3(t) = \Delta u(t)^T\, W_{\Delta u}\, \Delta u(t), \tag{15}$$

where $W_{\Delta u}$ is a $m \times m$ weight matrix.

5. Main Results

5.1. Case Study Partitioning

This section presents the results of the application of Algorithm 1 for the partitioning of the Barcelona DWN into compositional subsystems. Algorithm 1 and auxiliary routines presented in Section 3.2 have been designed for any system. However, some particular features should be introduced depending on the considered case study and control law in order to obtain an suitable decomposition. More precisely, the graph of the Barcelona DWN, shown in Figure 2, has been derived from its mathematical model (12) under the following considerations:

- every tank, sector of consume, water source and node is considered as a vertex of the graph;
- every pump, valve and link with a sector of consume is considered as a graph edge.

In order to evaluate the partitioning results obtained from the application of Algorithm 1 and auxiliary routines to the Barcelona DWN, the following indexes are taken into account additionally to those introduced in Section 3:

- $\varepsilon \triangleq \sum_{i=1}^{k} \varepsilon_i$,
- $\overline{\varepsilon} \triangleq \frac{\varepsilon}{k}$ (arithmetic mean),
- $\sigma_\varphi^2 \triangleq \frac{1}{k} \sum_{i=1}^{k} (\varphi_i - \overline{\varphi})^2$,
- $\sigma_\varepsilon^2 \triangleq \frac{1}{k} \sum_{i=1}^{k} (\varepsilon_i - \overline{\varepsilon})^2$.

Table 1: Results for different partitioning approaches

Routine Combination	k	$\bar{\varphi}$	$\bar{\varepsilon}$	σ^2_φ	σ^2_ε	ε
1	17	10.59	3.76	53.88	25.32	64
2	13	6.30	4.15	21.39	27.80	54
3	10	8.20	5.10	31.73	32.76	52
4	6	13.67	6.33	**14.88**	25.22	**38**

Remark 5.1. *Notice that although ε is not directly related with the number of shared edges between subgraphs obtained by using Algorithm 1, this index gives an indirect idea about their level of interconnection. Recall that the objective of the partitioning algorithm is the minimisation of indexes σ^2_φ, ε, and ε_i (for $i = 1, 2, \ldots, k$) to obtain a graph decomposition as less interconnected as possible and with similar number of vertices for each subgraph (internal weight).*

Table 1 summarises the partitioning results obtained applying Algorithm 1 (A1) combined with the auxiliary routine/filters presented in Section 3.2 performing the following combinations:

1. No auxiliary routines are considered.
2. A1 and Pre-filtering (Pre-F) routine only.
3. A1 in addition to Pre-F and Post-filtering (Post-F) routines.
4. A1 in addition to Pre-F, Post-F and Anti-Oscillation (AO) routines.

This distinction has been done in order to understand how the proposed routines affect the partitioning results.

Using only the Algorithm 1, the resultant partitioning P is comprised by 17 subgraphs. Many of them are small and cannot be merged since their neighbour subgraphs have internal weights with values quite close to $\bar{\varphi}$ (see Section 3.2). Moreover, there are several vertices with $\omega = 1$, which correspond to network water sources and demands, leading to unnecessarily difficult algorithm computations due to sizes of the resultant subgraphs (in terms of internal weight). By employing the Pre-F routine, the previous problems are fixed and Algorithm 1 produces 13 subgraphs (see Table 1). Additionally, if the refining routine embedded within Algorithm 1 is complemented with the Post-F routine, setting $\delta = 2$, a partitioning with ten subgraphs is reached[5]. Finally, if the AO routine is also considered, setting the refining limit to $\rho = 250$, a partitioning with six subgraphs is now reached. According to Table 1, this last partitioning (Combination 4) satisfies the minimisation of the average of the internal weights for all resultant subgraphs as well as the interconnection degree between subgraph measured through ε. It is important to highlight that the proposed partitioning approach automatically determines the final number of partitions k (six for this case) when the conditions 3 and 4 of Problem 1 are fulfilled (see Remark 2.2). The tuning parameters δ and ρ also influence in the obtained value of k.

[5]Notice that increasing the parameter δ implies that $\sigma^2_{\varepsilon_i}$ becomes bigger.

Table 2: Dimension comparison of the DWN subsystems

Subsystem	Tanks	Actuators	Demands	Nodes
1	13	36	20	5
2	11	11	11	0
3	13	22	20	3
4	9	16	12	2
5	6	10	8	2
6	15	26	17	3
Total	67	121	88	15

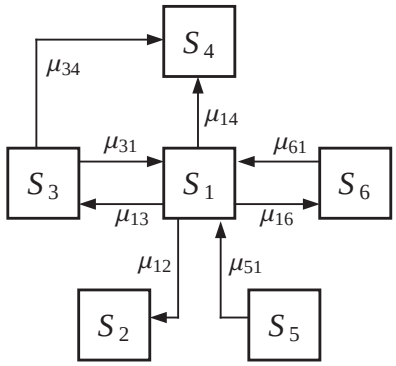

Figure 4: Network subsystems S_i and their sets of shared connections μ_{ij}

Notice that each subgraph of the final decomposition corresponds to a subsystem of the Barcelona DWN with the number of elements presented in Table 2. Figure 3 shows, in different colours, the obtained subsystems of Barcelona DWN. Moreover, Figure 4 schematically depicts the resultant subsystems S_i, for $i \in \{1, \ldots, 6\}$, and the sets μ_{ij} of shared links between the network subsystems corresponding to the control inputs u (manipulated flows, see (12)), whose directionality is defined from S_i to S_j for $j \in \{1, \ldots, 6\}$, $i \neq j$. Table 3 collects the number of control inputs of each set μ_{ij}.

5.2. DMPC based on a Hierarchical-like Approach

5.2.1. Strategy Parametrisation

Using the Barcelona DWN decomposition obtained in previous section (corresponding to the routine combination 4 in Table 1), a DMPC strategy is implemented in order to manage the networked system. This strategy considers

- the dynamical system model in (12) split in 6 subsystems obtained by using the proposed partitioning approach;

Table 3: Dimensions of shared links μ_{ij}

Set	μ_{12}	μ_{13}	μ_{14}	μ_{16}	μ_{31}	μ_{34}	μ_{51}	μ_{61}
Number of u's	2	2	2	2	4	3	1	3

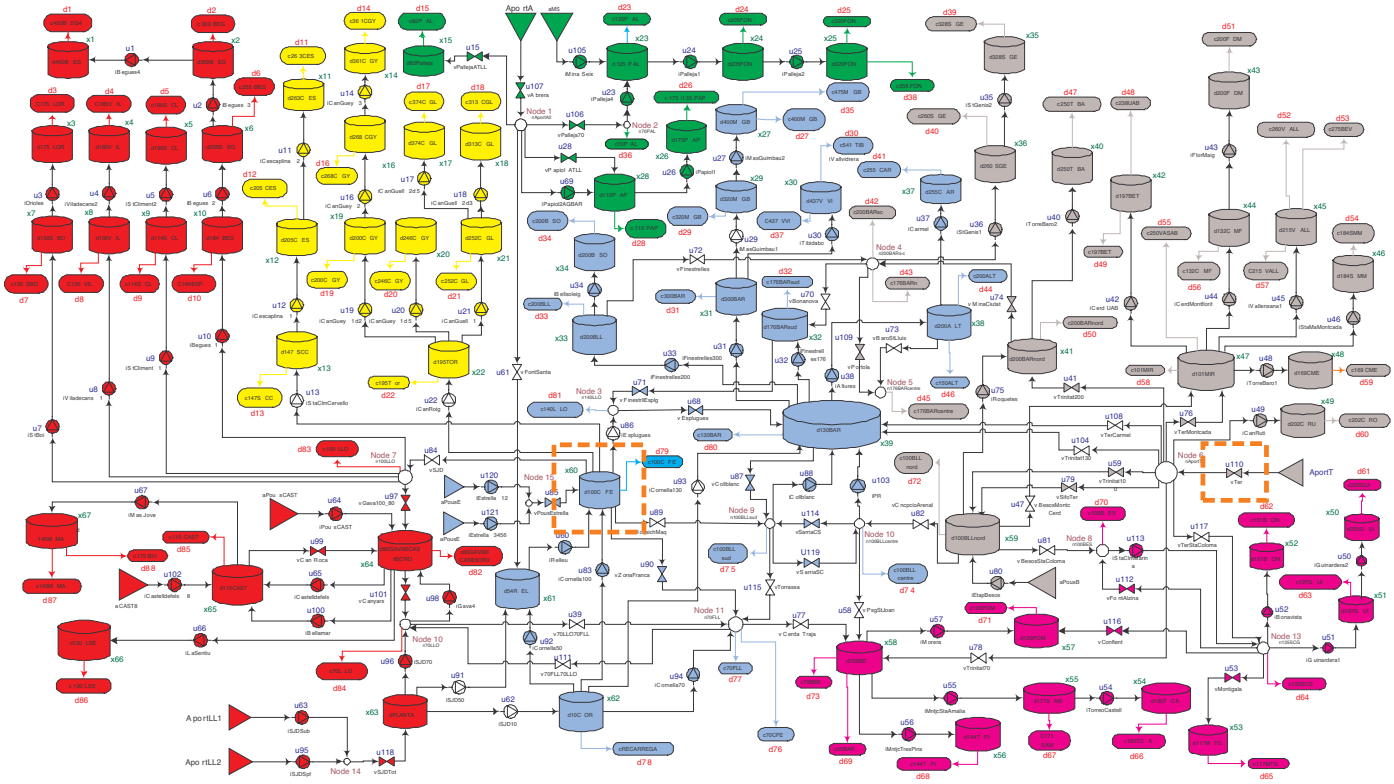

Figure 3: Definitive partition of the Barcelona DWN. The *key* elements are properly featured

- the physical constraints (9) and (10) for each subsystem;
- a demand forecasting algorithm (taken from [32, 31]); and
- a multi-objective cost function, expressed by using (13), (14), and (15) as

$$J(t) = \sum_{i=0}^{H_u-1} f_1(t+i|t) + \sum_{i=1}^{H_p} f_2(t+i|t) + \sum_{i=0}^{H_u-1} f_3(t+i|t), \quad (16)$$

where H_p and H_u correspond to the prediction and control horizons, respectively, index t represents the current time instant while index i represents the predicted time along H_p. In the case study of this paper, the prediction horizon is related to the 24-hours demand seasonality. Regarding the value of H_u, it has been set to be equal to H_p, following the criterion of the DWN management company.

In order to explain and discuss the implementation of the solution sequence for the considered hierarchical-like DMPC strategy, denote C_i as the MPC controller related to the subsystem S_i (for $i \in \{1, \ldots, 6\}$), and notice that, at this stage, μ_{ij} not only contains values of each component at time step t but also all values over H_u, i.e., if $\mu_{ij} = \{u_a, u_b, \ldots\}$, then[6]

$$\begin{aligned} u_a &\triangleq [u_a(t|t) \quad u_a(t+1|t) \quad \ldots \quad u_a(t+H_u-1|t)]^T, \\ u_b &\triangleq [u_b(t|t) \quad u_b(t+1|t) \quad \ldots \quad u_b(t+H_u-1|t)]^T, \quad (17) \\ &\vdots \end{aligned}$$

with $u_a(t+i|t)$ denoting the value of u_a at time step $t+i$ (over the control horizon) given t. Additionally, the following definition is introduced.

Definition 1 (Virtual demand). *Consider two subsystems S_1 and S_2, which share a set of manipulated flows μ_{12}. According to the notation employed in the paper, those flows come from S_1 to S_2. If the solution sequence of optimisation subrproblems — defined by the pre-established hierarchical order — determines that μ_{12} is computed by the MPC controller of S_1, then flows in μ_{12} are considered as virtual demands in the controller related to S_2 since their value are now imposed in the same way as the water demands.*

According to [2], the *pure* hierarchical control scheme determines a sequence of information distribution among the subsystems, where top-down communication is available from upper to lower level of the hierarchy. Note that, despite the subsystems coupling (given by the shared links), the main feature of

[6] With a slight abuse of the notation, the elements of vector u are denoted with the corresponding discrete-time dependence in order to differentiate the vector from its components.

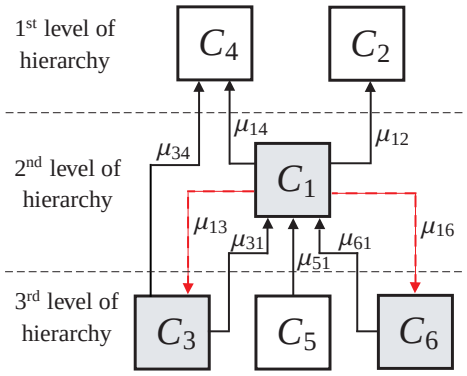

Figure 5: Hierarchy of MPC controllers C_i. Their solution sequence is top-down

the pure hierarchical control approach relies on the unidirectionality of the information flow between controllers.

Looking at Figure 5, where the directions of sets μ_{ij} are graphically shown, it is possible to realise that two of those sets, denoted by μ_{13} and μ_{16} (red dashed lines in the figure), break the mentioned unidirectional flow between MPC controllers. This fact implies that the standard hierarchical control scheme for partitioned LSS cannot be straight applied. To solve this situation and design a DMPC strategy, a hierarchical-like DMPC approach proposed in [18] has been considered and conveniently implemented over the partitioned system depicted in Figure 4. This strategy follows the hierarchical control philosophy and the sequential way of solving the optimisation subproblems of the corresponding MPC controllers but also considering the appearance of bidirectional information flows. For this purpose, additional constraints and heuristics are taken into account in order to cope with the feature of having the double direction in the flow of information between some of the controllers. In particular, Figure 5 shows the considered hierarchy for the case study of this paper, where controllers at the first level of hierarchy determine the values of variables shared with controllers in lower levels. Notice that Figure 5 also shows why the pure hierarchical control approach cannot be employed since the MPC controller related to the subsystem 1 shares bidirectional information with the controllers of S_2 and S_3.

Therefore, the solution sequence of the described hierarchical-like control problem for the complete Barcelona DWN at each time step $t \in \mathbb{Z}_{\geq 1}$ is the following:

- C_4 computes the control actions of S_4 and sets μ_{14} and μ_{34}.
- In parallel, C_2 computes the control actions of S_2 and the set μ_{12}.
- C_1 computes the control actions of S_1 and sets μ_{31}, μ_{51}, and μ_{61}. Sets μ_{12}, μ_{13}, μ_{14}, and μ_{16} are considered as sets of *virtual demands* within the controller C_1.
- C_5 computes the control actions of S_5 considering μ_{51} as a set of virtual demands.
- C_3 computes the control actions of S_3 considering μ_{31} and μ_{34} as sets of virtual demands. C_3 also computes the set μ_{13} to be used as a set of virtual demands for C_1 at time step $t+1$.
- C_6 computes the control actions of S_6 considering μ_{61} as a set of virtual demands. C_6 also computes μ_{16} to be used as a set of virtual demands for C_1 at time step $t+1$.

Remark 5.2. *Notice that in the solution sequence of the considered DMPC scheme, at the first time step ($t = 1$), the initial values of the control actions belonging to sets μ_{13} and μ_{16} are not available. Those values can be obtained by solving a constraint satisfaction problem (CSP) defined by the models and constraints of subsystems S_1, S_3 and S_6 (shaded blocks in Figure 5) through the algorithm proposed in [33]. The solution of this CSP provides feasible control actions for sets μ_{13} and μ_{16}, which allows starting the solution sequence described above. For subsequent time steps, values of μ_{13} and μ_{16} take values computed by C_3 and C_6, respectively, in the previous time step, i.e., the elements belonging to those sets at time step t are now assigned as (see* (17)*)*

$$u = \begin{bmatrix} u(t+1|t-1) \\ \vdots \\ u(t+H_u-1|t-1) \\ u(t+H_u-1|t-1) \end{bmatrix}.$$

5.2.2. Simulation Results

The results obtained by using this DMPC strategy are compared with those obtained employing a centralised MPC approach. Two scenarios corresponding to different prioritisations of the control objectives have been considered for the performance comparison of the MPC strategies:

- *Scenario 1:* $\Psi = (0.7, 0.2, 0.1)$,
- *Scenario 2:* $\Psi = (0.6, 0.2, 0.2)$,

where $\Psi = (\psi_e, \psi_x, \psi_{\Delta t})$ represents the 3-tuple of weights associated to the weight matrices $W_e \triangleq \psi_e I$, $W_x \triangleq \psi_x I$, $W_{\Delta u} \triangleq \psi_{\Delta u} I$ at the normalised functions (13), (14), and (15), respectively[7]. Notice that, given the employed normalisation of the control objective terms in the cost function (16), the sum of ψ_i, for $i \in \{e, x, \Delta u\}$, should be 1. The tuning scenarios are chosen in a way that the highest priority objective is the economic cost (see Section 4.3), which should be minimised while maintaining a similar rate of the safety volume and control action smoothness terms.

All results have been obtained considering real demands of four days (with 1 hour sampling time), with initial volumes in tanks set to 40% of their maximum volume, $H_p = H_u = 24$, and the safety volume parameter β set to 0.8. All simulations have been performed in MATLAB® 7.1 implementations running on an Intel® Core™2, 2.4 GHz machine with 4Gb RAM.

Table 4 summarises the obtained control results in terms of performance (economical cost) and computational burden over

[7] Matrix I denotes the identity matrix of suitable dimensions.

Table 4: Computation time and performance comparisons

Index	Scenario 1		Scenario 2	
	CMPC	DMPC	CMPC	DMPC
Water Cost	138.37	189.45	137.05	188.81
Electric Cost	92.73	68.44	87.43	69.91
Total Cost	231.10	257.89	224.48	258.72
CPU time	1143	537	1127	560

four days. The indexes representing costs are given in economic units (e.u.) instead of Euro due to confidentiality restrictions. Computation times are given in seconds.

From Table 4, it can be noticed the increment of the total costs of operation when using the DMPC strategy, what implies a loss of performance of about 15%. This loss of performance is obtained because the DMPC strategy does not take into account in a proper way the water costs related to external water sources since it is a global objective. On the other hand, DMPC controllers are mainly focused on the reduction of pumping costs (local objective) within each subsystem. By contrast, the information of water costs is properly managed for the CMPC controller by optimising it but at the price of *moving* more water inside the network. This leads to an increment in the electric costs (the water transportation cost) when CMPC controller is used. Therefore, despite the DMPC approach inevitably leads to a loss of performance, the benefits in terms of time and computational burden are significant enough, what makes it suitable for real-time implementation purposes. Notice that in this particular application, the CMPC could also satisfy the real-time constraint since the control sampling time is 1 hour. Thus, the main motivation for using DMPC in this application would not be the improvement in computation but the scalability and the potential adaptability easiness facing network changes that could occur. In fact, according to discussions with the AGBAR company, the main reason for using a DMPC approach in the case study of this paper, additionally to the easier maintenance of the (sub)system models, is that it allows replacing the current legacy control in multiple steps, where the DMPC is implemented on a selected network part only at each step. This ability is important for practical application and maintenance, which allows moving some part of the network to the current legacy control when some malfunction/fault is detected without stopping the supervisory MPC controller.

Regarding the closed-loop behaviour of the network, Figures 6 and 7 show the flow through a water supply valve and a the volume of a *key* tank, respectively (see the highlighted elements in Figure 3), for both predictive control strategies. Notice in Figure 6 that the behaviour of the volume is qualitatively equivalent for both strategies since the filling and emptying processes of the associated tank follows the demand evolution. On the other hand, notice in Figure 7 that the water inflow from this source is greater when DMPC is implemented. As discussed before, DMPC strategy makes that the water of each subsystem is supplied by its own sources, reducing the water transportation within the network. Hence, this source is providing almost all the water that this subsystem needs in contrast to the CMPC case, where the water was moved from other network locations (due to its cheaper price). This fact explains why the DMPC strategy yields a suboptimal solution compared with the CMPC counterpart. This degree of suboptimality is inherent to the followed hierarchical approach since each controller is mainly focused on optimising the control objectives related to the subnetwork that is controlling. A further improvement of the considered DMPC strategy would consist in adding some improved coordination mechanism in the control objective of each local MPC controller in order to enforce the fulfilment of global control objectives (see, e.g., [34]). This improvement will allow to take into account the economic costs in a global way.

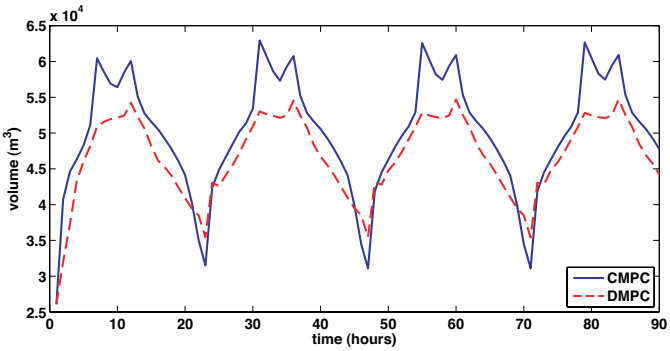

Figure 6: Resultant volume related to a *key* tank within the DNW

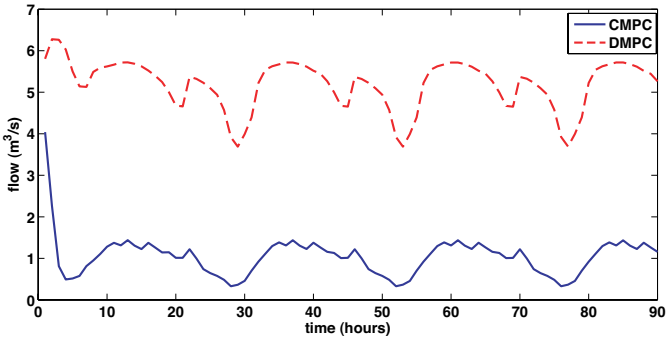

Figure 7: Computed flow related to a supply water valve

5.2.3. *Relation between the System Decomposition and the DMPC Performance*

The relation between a given system decomposition obtained when using Algorithm 1 and the performance of the employed DMPC scheme is discussed in this section. Table 4 presents performance indexes in terms of economic costs, what allows comparing the results obtained with a CMPC and the used DMPC controllers. This fact implies the evaluation of (13) with the control inputs computed by the decentralised controllers. Notice that, if a DMPC scheme is considered, the vector of input variables is given by

$$u = [u_{\text{INT}} \quad u_{\text{SHD}}]^T,$$

where $u_{\text{INT}} \in \mathbb{U}_{\text{INT}}$ denotes the vector of those control inputs that belong to one and only one subsystem S_i (for $i \in \{1, \ldots, k\}$), and $u_{\text{SHD}} \in \mathbb{U}_{\text{SHD}}$ denotes the vector of shared control inputs between subsystems. Moreover, $\mathbb{U} = \mathbb{U}_{\text{INT}} \cup \mathbb{U}_{\text{SHD}}$. Hence, since the optimisation variables correspond to the system control inputs, any of the performance index described in Section 4.3 —such as the economic cost (13)— can be written without lost of generality as[8]

$$f(u) = l_1(u_{\text{INT}}) + l_2(u_{\text{SHD}}), \tag{18}$$

where $l_1 : \mathbb{U}_{\text{INT}} \mapsto \mathbb{U}$ and $l_2 : \mathbb{U}_{\text{SHD}} \mapsto \mathbb{U}$ denote the corresponding mapping functions according to each particular case. Notice the straight relation between the size of the vector of shared controls u_{SHD} and the partitioning index ε. This latter is associated to the cut size of the entire system graph what, in turn, measures the number of interconnections (shared controls) between subsystems. Also notice that the suboptimal performance degree of the considered DMPC strategy is mainly related to the second term of (18), i.e., to the number of shared control inputs. The influence of this term decreases as ε tends to zero. At this point, two cases can be stated:

- The case $\varepsilon = 0$ and $k = 1$ corresponds to a CMPC strategy with $l_2(u_{\text{SHD}}) = 0$.
- The case $\varepsilon = 0$ and $k > 1$ implies that the resultant subsystems are decoupled since they do not share any control input. Therefore, the performance of the DMPC is optimal since $l_2(u_{\text{SHD}}) = 0$ (see [10, 35]).

This fact justifies to look for a system decomposition with a small ε (see Remark 5.1), i.e., with less shared links (control inputs) between its subsystems since it implies a less suboptimal performance of the considered DMPC strategy.

6. Conclusions

This paper has proposed a graph-theory-based algorithm for the automatic partitioning of large-scale systems into subsystems intended to be applied along with a decentralised model predictive control strategy. The algorithm transforms the dynamical model of the given system into a graph representation. Once the equivalent graph has been obtained, the problem of graph partitioning is then solved. The resultant partitions are composed of a set of non-overlapping subgraphs such that their sizes, in terms of number of vertices, are similar and the number of edges connecting them is minimal. To achieve this goal the algorithm applied a set of procedures based on identifying the highly-connected subgraphs with balanced number of internal and external connections. Some additional pre-filtering and post-filtering routines are also needed to be included to reduce the number of obtained subsystems. The performance of the proposed decomposition approach has been assessed in a real case study based on the Barcelona drinking water network. An study of the effect of auxiliary routines on the basic partitioning algorithm has also been included showing the benefits of their use. Promising control results have been obtained using a hierarchical-like DMPC approach, which makes use of this partitioning. A comparison with a CMPC approach show that the level of sub-optimality in economic costs is acceptable considering the resultant reduction in computational burden.

As future research, further improvements of the proposed partitioning algorithm, considering particular specifications imposed by the decentralised control strategies, should be added as well as different ways of treating the sets of shared control actions between subsystems and weighting policies for their consideration by the controllers. Moreover, the hierarchical-like DMPC strategy considered in this paper, which addresses the loops between hierarchical levels in a heuristic way, might be further investigated in order to evaluate the introduced degree of suboptimality as well as how feasibility and stability features are preserved.

[8] The dependence of t is omitted for compactness.

Acknowledgements

This work has been supported by Spanish research project WATMAN (CICYT DPI2009-13744) of the Science and Technology Ministry, the *Juan de la Cierva* Research Programme (ref. JCI-2008-2438), the DGR of Generalitat de Catalunya (SAC group Ref. 2009/SGR/1491) and the EU project WIDE (FP7-IST-224168). The authors specially thank the support received from AGBAR in the case study of this paper and the interesting suggestions and comments from the reviewers of the manuscript.

References

[1] J. Lunze, Feedback Control of Large-Scale Systems, Prentice Hall, Great Britain, 1992.

[2] D. Šiljak, Decentralized control of complex systems, Academic Press, 1991.

[3] J. Maciejowski, Predictive Control with Constraints, Prentice Hall, Great Britain, 2002.

[4] M. Brdys, B. Ulanicki, Operational Control of Water Systems: Structures, algorithms and applications, Prentice Hall International, UK, 1994.

[5] M. Marinaki, M. Papageorgiou, Optimal Real-time Control of Sewer Networks, Springer, Secaucus, NJ (USA), 2005.

[6] P. V. Overloop, Model Predictive Control on Open Water Systems, Delft University Press, Delft, The Netherlands, 2006.

[7] R. Negenborn, Multi-agent model predictive control with applications to power networks, Ph.D. thesis, Delft University of Technology, Delft, The Netherlands (September 2008).

[8] HD-MPC, Hierarchical and distributed model predictive control of large-scale complex systems, Home page (2008).
URL `http://www.ict-hd-mpc.eu/`

[9] WIDE, Decentralized and wireless control of large-scale systems, Home page (2008).
URL `http://ist-wide.dii.unisi.it/`

[10] T. Keviczky, F. Borrelli, G. Balas, Decentralized receding horizon control for large scale dynamically decoupled systems, Automatica 42 (12) (2006) 2105–2115.

[11] W. B. Dunbar, Distributed receding horizon control of dynamically coupled nonlinear systems, IEEE Transactions of Automaic Control 52 (7) (2007) 1249–1263.

[12] J. B. Rawlings, B. T. Stewart, Coordinating multiple optimization-based controllers: New opportunities and challanges, Journal of Process Control 18 (9) (2008) 839–845.

[13] R. Negenborn, B. De Schutter, J. Hellendoorn, Multi-agent model predictive control for transportation networks: Serial vs. parallel schemes, Engineering Applications of Artificial Intelligence 21 (3) (2008) 353–366.
[14] A. N. Venkat, I. A. Hiskens, J. B. Rawlings, S. J. Wright, Distributed MPC strategies with application to power system automatic generation control, IEEE Tran on Control Systems Technology 16 (6) (2008) 1192–1206.
[15] R. Scattolini, Architectures for distributed and hierarchical Model Predictive Control: A review, Journal of Process Control 19 (5) (2009) 723–731.
[16] F. Li, W. Zhang, Q. Zhang, Graphs partitioning strategy for the topology design of industrial network, IET Communications 1 (6) (2007) 1104–1110.
[17] P. Fjallstrom, Algorithms for graph partitioning: A survey, Linkoping Electronic Articles in Computer and Information Science 3 (10).
[18] C. Ocampo-Martinez, V. Fambrini, D. Barcelli, V. Puig, Model predictive control of drinking water networks: A hierarchical and decentralized approach, in: Proceedings of the American Control Conference, Baltimore (USA), 2010.
[19] A. Zecevic, D. Šiljak, Balanced decompositions of sparse systems for multilevel parallel processing, IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications 41 (3) (1994) 220–233.
[20] A. I. Zečević, D. D. Šiljak, Control of Complex Systems: Structural Constraints and Uncertainty, Communications and Control Engineering, Springer, 2010.
[21] J. Bondy, U. Murty, Graph Theory, Vol. 244 of Graduate Series in Mathematics, Springer, 2008.
[22] T. Bui, B. Moon, Genetic algorithm and graph partitioning, IEEE Transactions on Computers 45 (7) (1996) 841–855.
[23] S. Dutt, New faster kernighan-lin-type graph-partitioning algorithms, in: Proceedings of the IEEE/ACM international conference on Computer-aided design, IEEE Computer Society Press, 1993, pp. 370–377.
[24] A. Gupta, Fast and effective algorithms for graph partitioning and sparse-matrix ordering, IBM Journal of Research and Development 41 (1) (1997) 171–183.
[25] M. Sezer, D. Šiljak, Nested ε-decompositions and clustering of complex systems, Automatica 22 (3) (1986) 321–331.
[26] D. Van den Bout, T. Miller III, Graph partitioning using annealed neural networks, in: Proceedings of the International Joint Conference on Neural Networks, IJCNN, Washington, D.C. (USA), 1989, pp. 521–528.
[27] K. Schloegel, G. Karypis, V. Kumar, Parallel multilevel algorithms for multi-constraint graph partitioning, in: Euro-Par 2000 Parallel Processing, Springer, pp. 296–310.
[28] B. Hendrickson, T. Kolda, Graph partitioning models for parallel computing, Parallel Computing 26 (12) (2000) 1519–1534.
[29] T. Cormen, Introduction to algorithms, The MIT press, 2001.
[30] L. Addario-Berry, K. Dalal, B. Reed, Degree-constrained subgraphs, Discrete Applied Mathematics 156 (7) (2008) 1168–1174.
[31] J. Quevedo, V. Puig, G. Cembrano, J. Blanch, Validation and reconstruction of flow meter data in the Barcelona water distribution network, Control Engineering Practice 11 (6) (2010) 640–651.
[32] C. Ocampo-Martinez, V. Puig, G. Cembrano, R. Creus, M. Minoves, Improving water management efficiency by using optimization-based control strategies: the Barcelona case study, Water Science & Technology: Water supply 9 (5) (2009) 565–575.
[33] L. Jaulin, M. Kieffer, O. Didrit, E. Walter, Applied Interval Analysis, with Examples in Parameter and State Estimation, Robust Control and Robotics, Springer-Verlag, London, 2001.
[34] A. Bemporad, C. Pascucci, C. Rocchi, Hierarchical and hybrid model predictive control of quadcopter air vehicles, in: Proceedings of the 3rd IFAC Conference on Analysis and Design of Hybrid Systems, Zaragoza (Spain), 2009, pp. 14–19.
[35] A. Alessio, A. Bemporad, Decentralized model predictive control of constrained linear systems, in: Proceedings of the European Control Conference, Kos, Greek, 2007, pp. 2813–2818.