Online Partitioning Method for Decentralized Control of Linear Switching Large-Scale Systems[☆]

Wicak Ananduta[a,*], Tomás Pippia[b], Carlos Ocampo-Martinez[a], Joris Sijs[b,c], Bart De Schutter[b]

[a] *Institut de Ròbotica i Informàtica Industrial, CSIC-UPC, Carrer Llorens i Artigas 4-6, 08028 Barcelona, Spain*
[b] *Delft Center for Systems and Control, Delft University of Technology, Delft, The Netherlands*
[c] *TNO Technical Sciences, Den Haag, The Netherlands*

Abstract

A novel partitioning approach for linear switching large-scale systems is presented. We assume that the modes of the switching system are unknown a priori but can be detected. We propose an online partitioning scheme that can partition the system when the mode switches, thus adapting the partition to the mode. Moreover, after the system has been partitioned, we apply a decentralized state-feedback control scheme to stabilize the system. We also apply a dwell time stability scheme to prove that the closed-loop system remains stable even after both the mode and partition changes. The proposed approach is illustrated by means of an automatic generation control problem related to frequency deviation regulation in a large-scale power network.

Keywords: System partitioning, large-scale systems, decentralized control, switching systems, automatic generation control

1. Introduction

Large-scale systems (LSSs) are systems in which the number of compositional elements are both large in number and geographically widespread [1–6]. Examples include water networks [2], traffic networks [3], and power networks [4]. Moreover, LSSs can also be time-varying, in the sense that some characteristics or parameters, such as their topologies, could be not constant along time.

Due to the large amount of data and elements in the network, control of LSSs is not a trivial task [7]. Although in small-sized plants a centralized controller can make the closed-loop system achieve a suitable performance, in LSSs a centralized controller would have to face many issues related to the amount of data and distance between elements [8, 9]. One of the problems is related to the communication between elements of the network, since the distance between them might cause problems such as delays and packet loss [7, 9, 10]. This fact holds in cases in which a centralized controller would have to collect information

[☆]This work has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 675318 (INCITE). *(Wicak Ananduta and Tomás Pippia are co-first authors.)*

[*]Corresponding author
Email addresses: wananduta@iri.upc.edu (Wicak Ananduta), t.m.pippia@tudelft.nl (Tomás Pippia), cocampo@iri.upc.edu (Carlos Ocampo-Martinez), j.sijs@tudelft.nl (Joris Sijs), b.deschutter@tudelft.nl (Bart De Schutter)

about the states from many or all the other nodes, e.g., state feedback control or model predictive control. Moreover, for some optimization-based control strategies, the computational complexity arising from the centralized control of the LSS might make the problem too difficult to solve in a limited amount of time. An idea to overcome these problems is to partition the system into subsystems and apply a non-centralized controller, which can be either decentralized or distributed. In both cases, some problems of the centralized scheme could be overcome, since the control input is computed and applied locally. In decentralized control approaches, controllers do not exchange information amongst themselves and they apply a control input that does not take directly into account the coupling between different subsystems [8, 11, 12]. As one could expect, this strategy works better when the coupling between subsystems is weak. On the other hand, distributed control strategies consider that a communication infrastructure is present in the system and thus the different subsystems can exchange information amongst themselves [9, 10]. This information can be either related to the local state, or the local control action, or to both. Therefore, the local controllers can include extra information into their control problem. In both cases, the communication flow and the computational complexity per controller are reduced, since the LSS control problem is split into smaller control problems among several subsystems.

Prior to applying a non-centralized controller, partitioning or decomposition of the LSS into smaller subsystems is required. Some early works that propose an automatic system decomposition approach were published in the 1980s, e.g., [13, 14]. In these articles, the system is described as a graph and the partitioning objective is to minimize coupling between the resulting subsystems. Moreover, recent papers, e.g. [15–18] consider system decomposition as a graph partitioning problem. In this regards, the methods that are proposed in the aforementioned papers, can be classified into three broad classes: global methods, which take a graph as their input and produce a partition, e.g., spectral bisection methods [15, 16]; local improvement methods, which refine an initial partition [17]; and multi-step methods, which combine a simple global method and a local improvement method [18] in order to obtain a compromise between the computational burden and the quality of the solution.

However, to the best of our knowledge, little or no interest has been given to partitioning of time-varying systems. In the literature, the partitioning procedures are considered as an offline task that is carried out only once, before applying a non-centralized control approach. This fact could lead however to instability when the system under control is time-varying. Indeed, as discussed later in Example 1, since a change in the dynamics implies, in general, a change in the couplings between subsystems, a control scheme based on a previous description of the system might not be able to stabilize the system under control after that change.

We consider linear switching LSSs in this paper. We assume that, at certain moments in time, the description of the current system changes and the dynamics in the state space are described by a new (A, B) pair. Note that since we assume that we cannot control the switching sequence of the system towards different modes, we have a *switching* system, whereas in *switched* systems one can arbitrarily choose the switching sequence [19]. When studying stability of switching systems, one has to take directly into account

the switchings of the system and guaranteeing stability of each mode is not enough to guarantee stability of the overall system [20, 21]. Indeed, as shown in [22], switching between two asymptotically stable modes might result in a divergent trajectory; on the other hand, switching between two unstable modes can result in a stabilizing trajectory. Many results have been proposed for proving stability of switching systems [20]. In the current paper, we focus in particular on discrete-time switching systems, and therefore we build upon the results from [23, 24], which are based on dwell time, i.e., the amount of time during which each mode is active. These results pertain to switching systems in which the switchings are not so frequent and thus they are called *slowly* switching systems [20]. Note that [20] reviews stability and stabilization of switching systems. More recent results and related work on this topic include [25–27]. The work reported in [25] discusses stability for continuous-time positive systems with minimum dwell-time constraint; [26] deals with continuous-time nonlinear systems, in which some modes might be unstable, and uses a Takagi-Sugeno fuzzy modeling approach; [27] discusses switched positive fractional-order systems and proposes state-dependent switching signals. For what concerns dwell time schemes for discrete time systems, some works, e.g. [28–31], have been proposed after [23, 24]. However, since we consider linear discrete-time autonomous switching systems without delays, the results provided in [23, 24] are suitable in our case. For instance, [28] considers minimum dwell time, instead of *average* dwell time. In [29] the focus is on identifying a class of discrete-time switched signals that are globally asimptotically stable, therefore the paper deals with switched systems, and thus mainly the switching signals, while we consider *switching* systems. The works [30, 31] focus on input-to-state stability for nonlinear switching systems but our focus is on linear autonomous switching systems.

In this paper, we deal with decentralized control design of linear switching LSSs and the main contribution is twofold. Firstly, we propose an online partitioning method that is suitable for linear switching LSSs. Furthermore, we also provide convergence guarantees for the proposed partitioning algorithm. The proposed partitioning approach is inspired by [17, 18] and consists of two steps: an initial partitioning algorithm and a refining step. Differently from the multi-step method presented in [18], we prespecify the number of subsystems and already group together highly coupled components in the same subsystem in the initial partitioning procedure, so that the outcome of the initial partitioning procedure provides a warm start for the refining step. Secondly, we show that a decentralized state feedback control scheme for LSSs and stability results on switching systems under the average dwell time condition can be combined to stabilize the system. The overall proposed control scheme consists of a central coordinator and decentralized controllers. The central coordinator adjusts the partition and the decentralized state feedback gains in response to the switching nature of the system while the decentralized controllers stabilize the overall system via a decentralized state feedback scheme.

The structure of the paper is as follows. In Section 2, we provide a description of the problem that we consider. In Section 3, we present our partitioning algorithm. Sections 4 and 5 are devoted to the adopted decentralized state-feedback control scheme and to the stability analysis, respectively. In Section

6, we explain the overall control scheme and how the online partitioning procedure is carried out. We apply our proposed approach to an automatic generation control problem in Section 7 and lastly we provide some concluding remarks in Section 8.

Notation. We use calligraphic letters to denote sets, e.g., $\mathcal{P}$. The set cardinality and the 2-norm operators are denoted by $|\cdot|$ and $\|\cdot\|_2$, respectively. We use bold math symbols, e.g., $\boldsymbol{x}$, $\boldsymbol{A}$, for the centralized system. By $\dim(\cdot)$ we denote the dimension of a vector. Moreover, $\mathbb{R}_{>a}$ denotes all real numbers in the set $\{b : b > a,\ b, a \in \mathbb{R}\}$. A similar definition can be used for the non-strict inequality case. For vectors v_i with $i \in \mathcal{L} = \{l_1, \ldots, l_{|\mathcal{L}|}\}$, the operator $[v_i^\top]_{i\in\mathcal{L}}$ denotes the column-wise concatenation, i.e., $[v_i^\top]_{i\in\mathcal{L}} = [v_{l_1}^\top, \cdots, v_{l_{|\mathcal{L}|}}^\top]$. For matrices $M_{ij} \in \mathbb{R}^{n_i\times n_j}$ with $\mathcal{L} = \{l_1, \ldots, l_{|\mathcal{L}|}\}$ and $(i,j) \in \mathcal{L}\times\mathcal{L}$, the operator $[M_{ij}]_{(i,j)\in\mathcal{L}\times\mathcal{L}}$ denotes the matrix-wise concatenation, i.e.,

$$[M_{ij}]_{(i,j)\in\mathcal{L}\times\mathcal{L}} = \begin{bmatrix} M_{l_1 l_1} & \cdots & M_{l_1 l_{|\mathcal{L}|}} \\ \vdots & \ddots & \vdots \\ M_{l_{|\mathcal{L}|} l_1} & \cdots & M_{l_{|\mathcal{L}|} l_{|\mathcal{L}|}} \end{bmatrix}.$$

Finally, discrete-time instants are denoted by k.

2. Problem Statement

Consider a linear switching LSS that can be represented as a directed graph $\mathcal{G}(k) = (\mathcal{V}, \mathcal{E}(k))$, where $\mathcal{V} = \{1, 2, \ldots, |\mathcal{V}|\}$ denotes the set of components (vertices) and $\mathcal{E}(k) \subseteq \mathcal{V}\times\mathcal{V}$ denotes the set of edges that describes the interaction of the components among each other, i.e., edge $(j,i) \in \mathcal{E}(k)$ indicates that component j influences the dynamics of component i. Furthermore, the components of the network can be divided into two sets, which are denoted by $\mathcal{V}_x$ and $\mathcal{V}_u$, i.e., $\mathcal{V} = \mathcal{V}_x \cup \mathcal{V}_u$ and $\mathcal{V}_x \cap \mathcal{V}_u = \emptyset$. The set $\mathcal{V}_u$ contains all the input components, while $\mathcal{V}_x$ consists of all components that have dynamics as follows:

$$x_i(k+1) = \sum_{j\in\mathcal{V}_x} A_{ij}(k)x_j(k) + \sum_{j\in\mathcal{V}_u} B_{ij}(k)u_j(k),\ \forall i \in \mathcal{V}_x, \tag{1}$$

where $x_i \in \mathbb{R}^{n_i}$, denotes the state vector of component i and $u_j \in \mathbb{R}^{m_j}$, denotes the input from the components in $\mathcal{V}_u$. Additionally, for each $i \in \mathcal{V}_x$, $A_{ij} \in \mathbb{R}^{n_i\times n_j}$, and $B_{ij} \in \mathbb{R}^{n_i\times m_j}$, are the state-space matrices, where $\|A_{ij}\|_2 \neq 0$, if and only if $(i,j) \in \mathcal{E}(k)$, i.e. if and only if there is an edge between vertices i and j, and, similarly, $\|B_{ij}\|_2 \neq 0$, if and only if $(i,j) \in \mathcal{E}(k)$. Therefore, the dynamics of the overall system can be written as follows:

$$\boldsymbol{x}(k+1) = \boldsymbol{A}(k)\boldsymbol{x}(k) + \boldsymbol{B}(k)\boldsymbol{u}(k), \tag{2}$$

where $\boldsymbol{x}(k) = \left[x_i^\top(k)\right]_{i\in\mathcal{V}_x}^\top \in \mathbb{R}^n$ is the state of the overall system, $\boldsymbol{u}(k) = \left[u_i^\top(k)\right]_{i\in\mathcal{V}_u}^\top \in \mathbb{R}^m$ is the input of the overall system, $\boldsymbol{A}(k) = [A_{ij}(k)]_{(i,j)\in\mathcal{V}_x\times\mathcal{V}_x} \in \mathbb{R}^{n\times n}$, $\boldsymbol{B}(k) = [B_{ij}(k)]_{(i,j)\in\mathcal{V}_x\times\mathcal{V}_u} \in \mathbb{R}^{n\times m}$, $n = \sum_{i\in\mathcal{V}_x} n_i$, and $m = \sum_{i\in\mathcal{V}_u} m_i$. An example can be found in Figure 1.

We assume that the system represented by (2) belongs to the class of linear switching systems.

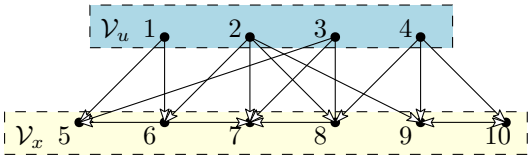


Figure 1: An illustration of a system with $|\mathcal{V}|= 10$. The set $\mathcal{V}_u = \{1, 2, 3, 4\}$ and $\mathcal{V}_x = \{5, 6, 7, 8, 9, 10\}$. The dots represent the components and the arrows represent the edges.

Definition 1. *A linear switching system is a system with dynamics given by*

$$\boldsymbol{x}(k+1) = \boldsymbol{A}_{\sigma(k)}\boldsymbol{x}(k) + \boldsymbol{B}_{\sigma(k)}\boldsymbol{u}(k),$$

where $\sigma(k)$ is a piecewise constant function called switching signal *that takes discrete values and associates to each time step k a different mode.* ♢

Remark 1. *Note that since we consider the class of switching systems, i.e., those under arbitrary switching signals, the paper does not concern the design of the switching signals $\sigma(k)$.* ♢

Although all the $(\boldsymbol{A}(k), \boldsymbol{B}(k))$ pairs of the system are unknown a priori, we assume that a change in the pair can be detected instantly. Note that the detection method is out of scope of this paper. Moreover, we assume that each mode is not active only once but it has a recurrent behavior. In this paper, we address the problem of stabilizing such systems with a decentralized state feedback control approach. Consider the discrete-time process described in (2). The state-feedback control law is obtained by applying the input $\boldsymbol{u}(k) = -\boldsymbol{K}(k)\boldsymbol{x}(k)$ to the system, where $\boldsymbol{K}(k) \in \mathbb{R}^{m\times n}$ is a time-varying gain matrix, obtaining the overall law $\boldsymbol{x}(k+1) = (\boldsymbol{A}(k) - \boldsymbol{B}(k)\boldsymbol{K}(k))\boldsymbol{x}(k)$. The matrix $\boldsymbol{A}(k)$ might not have asymptotically stable eigenvalues, but $\boldsymbol{K}(k)$ can be computed such that the final matrix $\boldsymbol{A}(k) - \boldsymbol{B}(k)\boldsymbol{K}(k)$ is asymptotically stable for all k.

In a decentralized control scheme, the system must be partitioned into several subsystems, to which local controllers are assigned. In this regard, the system must be partitioned such that the coupling between subsystems is minimized. Since the system is time-varying, the partition must also be adapted so that a stabilizing decentralized state-feedback controller can be designed. Example 1 shows the importance of changing partition when the mode of the system changes.

Example 1. *Consider a simple switching LTI system with two states ($x_1, x_2 \in \mathbb{R}$), two inputs ($u_1, u_2 \in \mathbb{R}$), and two modes (denoted by M_1 and M_2). The system has the following dynamics:*

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & a \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix} + \boldsymbol{B}(k) \begin{bmatrix} u_1(k) \\ u_2(k) \end{bmatrix},$$

where the matrix $\boldsymbol{A}(k) = \begin{bmatrix} a & 0 \\ 0 & a \end{bmatrix}$ is constant with $a \in \mathbb{R}_{>1}$, while the matrix $\boldsymbol{B}(k) = \begin{bmatrix} 1 & b \\ b & 1 \end{bmatrix}$, for mode M_1, and $\boldsymbol{B}(k) = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ for mode M_2, where $0 < b < 1/a$. Furthermore, consider that when $0 \le k \le k_1$, mode

M_1 is active and the system is partitioned into two subsystems (denoted by subsystem i and j) as follows: subsystem i consists of x_1 and u_1 and its dynamics are $x_1(k+1) = ax_1(k) + u_1(k) + bu_2(k)$, while subsystem j consists of x_2 and u_2 and its dynamics are $x_2(k+1) = ax_2(k) + bu_1(k) + u_2(k)$. Thus, by considering decentralized state-feedback gains $K_i(k), K_j(k) \in \mathbb{R}$ for subsystem i and j, respectively, i.e., the control laws are $u_1(k) = -K_i(k)x_1(k)$ and $u_2(k) = -K_j(k)x_2(k)$, the resulting centralized closed-loop system is

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} a - K_i(k) & -bK_j(k) \\ -bK_i(k) & a - K_j(k) \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix},$$

where $K_i(k)$ and $K_j(k)$ can be chosen such that the closed-loop matrix is asymptotically stable, e.g., $K_i(k) = K_j(k) = a$. Now, consider that the system switches to mode M_2, at $k = k_1 + 1$. If we keep the same partition, implying the same control laws, the centralized closed-loop system becomes

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} a & -K_j(k) \\ -K_i(k) & a \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix},$$

which is not asymptotically stable for any $K_i(k), K_j(k) \in \mathbb{R}$ since $a > 1$, which implies one of the eigenvalues will be outside the unit circle. Therefore, the system is not stabilizable due to its structure and due to the control law that we selected. However, if we change the partition such that subsystem i consists of x_1 and u_2 and its control law is $u_2(k) = -K_i(k)x_1(k)$, while subsystem j consists of x_2 and u_1 and its control law is $u_1(k) = -K_j(k)x_2(k)$, the centralized closed-loop system becomes

$$\begin{bmatrix} x_1(k+1) \\ x_2(k+1) \end{bmatrix} = \begin{bmatrix} a - K_i(k) & 0 \\ 0 & a - K_j(k) \end{bmatrix} \begin{bmatrix} x_1(k) \\ x_2(k) \end{bmatrix},$$

implying that it is stabilizable by the decentralized state-feedback gains $K_i(k), K_j(k)$, e.g., the gains can be set to $K_i(k) = K_j(k) = c$, where $a - 1 < c < a + 1$, to obtain an asymptotically stable system. ♢

3. System Partitioning

In this section, a time-varying partitioning approach is proposed such that a decentralized state feedback control scheme can be designed for the LSS.

3.1. Formulation of the Partitioning Problem

The system is partitioned into p non-overlapping subsystems and the objectives of the partitioning are to minimize the coupling among subsystems and to balance the number of inputs and states in each subsystem. Minimal coupling among subsystems is desirable for decentralized control structures, as reported in [11, 14] and shown before in Example 1. Furthermore, the computational burden would be equally distributed among the local controllers when the number of inputs and states of all subsystems are similar.

We consider system partitioning as a graph partitioning problem [17, 18]. Let $\mathcal{P}(k) = \{\mathcal{V}_1(k), \mathcal{V}_2(k), \ldots, \mathcal{V}_p(k)\}$ be the partition of $\mathcal{G}(k)$, where $\mathcal{V}_\ell(k)$, for each $\ell \in \{1, 2, \ldots, p\}$, indicates the set of vertices of subsystem ℓ.

The partitioning problem at time step k can be stated as follows:

$$\underset{\mathcal{P}(k)}{\text{minimize}} \quad \sum_{\ell=1}^{p} f_{\text{c}}(\mathcal{V}_\ell(k)) \tag{3a}$$

$$\text{subject to} \quad \bigcup_{\ell=1}^{p} \mathcal{V}_\ell(k) = \mathcal{V}, \tag{3b}$$

$$\mathcal{V}_\ell(k) \cap \mathcal{V}_j(k) = \emptyset,\ \ell \neq j,\ \forall \ell, j \in \{1, \ldots, p\}, \tag{3c}$$

where the cost function $f_{\text{c}}(\mathcal{V}_\ell(k)),\ \forall \ell \in \{1, \ldots, p\}$ is formulated according to the partitioning objectives, i.e.,

$$f_{\text{c}}(\mathcal{V}_\ell(k)) = \alpha_1 f_{\text{wc}}(\mathcal{V}_\ell(k)) + \alpha_2 f_{\text{im}}(\mathcal{V}_\ell(k)), \tag{4}$$

where $f_{\text{wc}}(\mathcal{V}_\ell(k))$ denotes the weighted cut cost, i.e., the sum of the weights $(w_{ij}(k))$ of the edges that connect subsystem ℓ and other subsystems, as follows:

$$f_{\text{wc}}(\mathcal{V}_\ell(k)) = \sum_{i \in \mathcal{V}_\ell(k)} \sum_{j \in \mathcal{V} \setminus \mathcal{V}_\ell(k)} \left(w_{ij}(k) + w_{ji}(k)\right),$$

and $f_{\text{im}}(\mathcal{V}_\ell(k))$ denotes the internal imbalance cost, i.e., the cost imposed to ensure that the number of inputs and states in every subsystem is nearly the same, as follows:

$$f_{\text{im}}(\mathcal{V}_\ell(k)) = \left| |\mathcal{V}_{u,\ell}(k)| - \frac{|\mathcal{V}_u|}{p} \right| + \left| |\mathcal{V}_{x,\ell}(k)| - \frac{|\mathcal{V}_x|}{p} \right|,$$

in which $\mathcal{V}_{u,\ell}(k) = \mathcal{V}_\ell(k) \cap \mathcal{V}_u$ and $\mathcal{V}_{x,\ell}(k) = \mathcal{V}_\ell(k) \cap \mathcal{V}_x$. Note that the weight of the edge $w_{ij}(k) \in \mathbb{R}_{\geq 0}$, is defined as

$$w_{ij}(k) = \begin{cases} \|A_{i,j}(k)\|_2 / \|\boldsymbol{A}(k)\|_2, & \text{if } i, j \in \mathcal{V}_x, \\ \|B_{i,j}(k)\|_2 / \|\boldsymbol{B}(k)\|_2, & \text{if } i \in \mathcal{V}_x,\ j \in \mathcal{V}_u, \\ 0, & \text{otherwise.} \end{cases}$$

Moreover, $\alpha_1, \alpha_2 \in \mathbb{R}_{>0}$ in (4), are tuning parameters that measure the importance of the objectives. Problem (3) is a combinatorial optimization problem that must be solved online. We propose a multi-step heuristic method that is able to find a local minimum of such a problem. The method is deeply explained in the next subsection.

3.2. Partitioning Algorithm

The partitioning algorithm is divided into two main steps. In the first step, an initial partition is obtained, and it is refined in the second step. In the initial partitioning, vertices that are highly coupled are grouped together, while maintaining a balanced number of vertices in each subsystem. The refining algorithm is a variation of the Kernighan-Lin algorithm [32] and similar to the method presented in [33, 34]. However, differently from our proposed approach, the method in [33, 34] considers balancing number of vertices as a constraint and is not particularly suitable for a system partitioning problem. Furthermore, while in [18] there are different steps taken for the two objectives, in our approach both objectives are merged in a single

cost function that we minimize. Additionally, unlike the method in [17] and as explained later in Proposition 1, by considering only one vertex at each iteration, we can ensure that the total cost is non-increasing among two consecutive iterations.

Initial partitioning algorithm. The number of subsystems, p, is determined such that $p \leq |\mathcal{V}_u|$ in order to ensure that at least one input is assigned to every subsystem. In this step, we want to have an initial partition $\mathcal{P}^{(0)}(k) = \{\mathcal{V}_1^{(0)}(k), \ldots, \mathcal{V}_p^{(0)}(k)\}$. To this end, the following algorithm is proposed:

1. Calculate the sum of the edge weights of all input vertices, as follows:

$$w_{\mathrm{s},i}(k) = \sum_{j \in \mathcal{N}_i(k)} w_{ji}(k), \quad \forall i \in \mathcal{V}_u,$$

   where $\mathcal{N}_i(k)$ is the set of neighbors of vertex i, i.e., $\mathcal{N}_i(k) = \{j : j \neq i, (i,j) \in \mathcal{E}(k)\}$.

2. Choose the centers of the subsystems, c_ℓ, for all $\ell \in \{1, 2, \ldots, p\}$, as the p input vertices that have the largest weight $w_{\mathrm{s},i}(k)$, and order them according to $w_{\mathrm{s},i}(k)$. At the end of this step, each subsystem has one input vertex.

3. Sequentially, starting from subsystem 1, find a vertex that is not assigned and has the highest coupling with one of the vertices in the evaluated subsystem, i.e., for subsystem ℓ,

$$\theta_\ell \in \arg\max_{\theta} \left\{ w_{i\theta}(k) + w_{\theta i}(k) \middle| \theta \in \mathcal{V} \backslash \left(\bigcup_{\ell=1}^{p} \mathcal{V}_\ell^{(0)}(k) \right) \right\},$$

   for all $i \in \mathcal{V}_\ell^{(0)}(k)$. The vertex θ_ℓ is randomly selected from the set of maximizers. Then, update $\mathcal{V}_\ell^{(0)}(k) \leftarrow \mathcal{V}_\ell^{(0)}(k) \cup \{\theta_\ell\}$. After θ_ℓ has been assigned to subsystem ℓ, we consider subsystem $\ell + 1$. If we reach subsystem p but there are still unassigned nodes, we go back to subsystem 1. Repeat this step until all of the vertices are assigned to a subsystem.

The complexity of the initial partitioning algorithm is polynomial, i.e., $\mathcal{O}((n+m)^2)$. Although the resulting partition has a balanced number of vertices at each subsystem, it is possible that the number of inputs and the number of states are not balanced. Furthermore, the weighted cut cost might also still be improved.

Refining algorithm. Consider the partition obtained from the initial partitioning algorithm, $\mathcal{P}^{(0)}(k)$. In the refining step, we improve the partition by moving one vertex at a time among subsystems. Denote subsystem ℓ as the subsystem that is selected to provide a proposal, i.e., one of its vertices that is considered to be moved to another subsystem. Furthermore, let $\hat{\mathcal{V}}_\ell(k)$ be the set of all vertices that belong to subsystem ℓ and that have at least an edge with a vertex that belongs to a different subsystem, i.e., $\hat{\mathcal{V}}_\ell(k) = \{i : (i,j) \in \mathcal{E}(k), i \in \mathcal{V}_\ell(k),\ j \in \mathcal{V} \backslash \mathcal{V}_\ell(k)\}$. Moreover, denote the iteration index by the superscript (r). The refining algorithm at the r^{th} iteration works as follows:

1. In order to select the subsystem ℓ, all subsystems are sorted in a descending order based on their costs, i.e., $f_{\mathrm{c}}(\mathcal{V}_j(k))$, for all $j \in \{1, 2, \ldots, p\}$. Then, the subsystem that has the highest cost is selected. Furthermore, once selected, it is discarded from the sorted list and when the sorted list is finally empty, a new sorted list is created using the same procedure.

2. Subsystem ℓ computes its current local cost $f_{\mathrm{c}}^{(r)}(\mathcal{V}_\ell(k))$ according to (4) and a proposal, i.e., a vertex that is offered to be moved to one of its neighbors, as follows:

$$\theta_\ell \in \arg \min_{\theta \in \hat{\mathcal{V}}_\ell(k)} \; f_{\mathrm{c}}(\mathcal{V}_\ell(k) \backslash \{\theta\}). \tag{5}$$

The vertex θ_ℓ is randomly selected from the set of minimizers according to (5). Notice that the set of minimizers in (5) might be empty, implying that $f_{\mathrm{c}}^{(r)}(\mathcal{V}_\ell(k)) f_{\mathrm{c}}(\mathcal{V}_\ell(k) \backslash \{\theta\})$, for any $\theta \in \hat{\mathcal{V}}_\ell(k)$. This means that no vertex is moved and the algorithm jumps to the next iteration. Otherwise, subsystem ℓ computes the expected local cost difference as follows:

$$\Delta f_{\mathrm{c},\ell}^{(r)}(k) = f_{\mathrm{c}}(\mathcal{V}_\ell(k) \backslash \{\theta_\ell\}) - f_{\mathrm{c}}^{(r)}(\mathcal{V}_\ell(k)). \tag{6}$$

Finally, it shares θ_ℓ and $\Delta f_{\mathrm{c},\ell}^{(r)}(k)$ with its neighbor subsystems, i.e. with all $j \in \mathcal{N}_{\mathcal{P},\ell}(k)$, where $\mathcal{N}_{\mathcal{P},\ell}(k) = \{j : (i, \theta_\ell) \in \mathcal{E}(k),\ i \in \mathcal{V}_j(k), j \neq \ell, j = 1, 2, \ldots, p\}$.

3. All the neighbors, i.e. all $j \in \mathcal{N}_{\mathcal{P},\ell}(k)$, compute the updated cost $f_{\mathrm{c}}(\mathcal{V}_\ell(k) \cup \{\theta_\ell\})$ if θ_ℓ is moved to them, according to (4). Then, the neighbors compute $\Delta f_{\mathrm{ct},j}^{(r)}(k)$ as follows:

$$\Delta f_{\mathrm{ct},j}^{(r)}(k) = f_{\mathrm{c}}\left(\mathcal{V}_\ell(k) \cup \{\theta_\ell\}\right) - f_{\mathrm{c}}^{(r)}(\mathcal{V}_j(k)) + \Delta f_{\mathrm{c},\ell}^{(r)}(k), \tag{7}$$

where $f_{\mathrm{c}}^{(r)}(\mathcal{V}_j(k))$ is the current cost, computed according to (4). Note that $\Delta f_{\mathrm{ct},j}^{(r)}(k)$ indicates the total cost difference when θ_ℓ moves from $\mathcal{V}_\ell(k)$ to $\mathcal{V}_j(k)$. Finally, the neighbors send $\Delta f_{\mathrm{ct},j}^{(r)}(k)$ to subsystem ℓ.

4. Subsystem ℓ decides to which subsystem it will send θ_ℓ as follows:

$$j^\star \in \arg \min_{j \in \mathcal{N}_{\mathcal{P},\ell}} \; \Delta f_{\mathrm{ct},j}^{(r)}(k). \tag{8}$$

If $\Delta f_{\mathrm{ct},j^\star}^{(r)}(k) > 0$, the algorithm jumps to the next iteration. Otherwise, the subsystem $j^\star$ is randomly selected from the set of minimizers according to (8).

5. The partition is updated as follows:

$$\mathcal{V}_\ell(k) \leftarrow \mathcal{V}_\ell(k) \backslash \{\theta_\ell\}, \tag{9}$$

$$\mathcal{V}_{j^\star}(k) \leftarrow \mathcal{V}_{j^\star}(k) \cup \{\theta_\ell\}. \tag{10}$$

The refining procedure, which is a polynomial time algorithm with respect to the number of vertices, i.e., $\mathcal{O}((n+m)^3)$, is summarized in Algorithm 1, where Θ denotes an auxiliary set that contains vertices that have at least one edge with a vertex in another subsystem, but that do not improve the total cost if they are moved. Furthermore, $\mathcal{V}_{\mathrm{so}}$ denotes an auxiliary ordered set that is used to select subsystem ℓ and c_{st} denotes the variable used to determine the stopping condition. Lastly, $\mathcal{V}_{\mathrm{so}}[1]$ denotes the first element of $\mathcal{V}_{\mathrm{so}}$.

Proposition 1. *The solution of the refining algorithm (Algorithm 1) converges to a local minimum. Furthermore, Algorithm 1 stops when a local minimum is reached.* ♢

Proof. In order to show the convergence, we will show that the total cost, denoted by $F^{(r)}(k) = \sum_{\ell=1}^{p} f_{\mathrm{c}}^{(r)}(\mathcal{V}_\ell(k))$, is non-increasing. Denote the initial partition at the r^{th} refining iteration by $\mathcal{V}_\ell^{(r)}$, for all $\ell \in \{1, 2, \ldots, p\}$. At the end of the r^{th} iteration, suppose that vertex θ_ℓ is moved from subsystem ℓ to subsystem $j^\star$. Thus, we have

$$\begin{aligned}\Delta F(k) =& F^{(r+1)}(k) - F^{(r)}(k)\\ =& f_{\mathrm{c}}(\mathcal{V}_{j^\star}^{(r+1)}(k)) - f_{\mathrm{c}}^{(r)}(\mathcal{V}_{j^\star}^{(r)}(k)) + f_{\mathrm{c}}(\mathcal{V}_{\ell}^{(r+1)}(k)) - f_{\mathrm{c}}^{(r)}(\mathcal{V}_{\ell}^{(r)}(k))\\ =& \Delta f_{\mathrm{ct},j^\star}^{(r)}(k) \leq 0.\end{aligned}$$

The second equality follows from the fact that only the cost of subsystems ℓ and $j^\star$ changes when vertex θ_ℓ is moved. The last inequality follows from the condition imposed in Step 3, in which vertex θ_ℓ is not moved if $\Delta f_{\mathrm{ct},j^\star}(k) > 0$. Note that when no vertex is moved, $F^{(r+1)}(k) - F^{(r)}(k) = 0$.

The proof of the second claim is as follows. Algorithm 1 stops if $c_{\mathrm{st}} = 2p - 1$. Furthermore, the counter c_{st} only increases if

$$f_{\mathrm{c}}^{(r)}(\mathcal{V}_\ell(k)) < f_{\mathrm{c}}(\mathcal{V}_\ell(k) \backslash \{\theta_\ell\}) \tag{11}$$

holds (see lines 10, 22, and 23). Moreover, if (11) is not satisfied, then c_{st} is reset to 0 (see line 12). Therefore, the algorithm only stops if (11) holds for at least $2p - 1$ consecutive iterations. Note that if (11) holds for all $\ell \in \{1, \ldots, p\}$, then a local minimum is reached since no more vertex from any subsystem is moved.

First, we show that condition (11) can be satisfied. Condition (11) holds if moving any vertex from $\hat{\mathcal{V}}_\ell$, defined in line 8, leads to the increasing of local cost $f_{\mathrm{c}}^{(r)}(\mathcal{V}_\ell(k))$ or if $\hat{\mathcal{V}}_\ell$ is empty. Furthermore, notice that θ_ℓ, which improves the local cost and is selected in line 9, might not be moved to a different subsystem because moving it might increase the total cost. In this case, to ensure the algorithm does not have an infinite loop, θ_ℓ will not be considered anymore for some iterations in the future (see lines 17 and 8). It also implies that, when a subsystem selects a potential vertex to be moved, it disregards vertices that have been identified in such a way that, if moved at the current iteration, the total cost will increase. Thus, line 8 might yield an empty $\hat{\mathcal{V}}_\ell$.

Now, we need to show that if the condition (11) holds for at least $2p - 1$ consecutive iterations, then we guarantee that (11) holds for all subsystems. This is a consequence of the way in which we select subsystem ℓ, i.e., lines 4 and 6. In the worst possible case, $2p - 1$ consecutive iterations are required to ensure all subsystems have been selected to propose θ_ℓ. Consider that at iteration r_1, $\mathcal{V}_{\mathrm{so}}^{(r_1)}$ is generated to select the subsystems for the next p iterations, starting from r_1. Suppose that subsystem p has the highest cost and, thus, is selected at r_1. Now, suppose that at iteration $r_1 + 1$, condition (11) is satisfied for the first time. Furthermore, (11) also holds for $r = r_1 + 2, \ldots, r_1 + p - 1$. Note that at $r_1 + p - 1$, all subsystems, but subsystem p, have been selected and have satisfied (11). At iteration $r_1 + p$, suppose that subsystem p has the lowest local cost and a new ordered set of $\mathcal{V}_{\mathrm{so}}$ is generated. Therefore, subsystem p will only be selected at iteration $r_1 + 2p - 1$, implying the necessity to evaluate $2p - 1$ consecutive iterations. For any other case, (11) must only hold for a number of consecutive iterations that is fewer than $2p - 1$. □

Algorithm 1 Refining Procedure

Input: $\mathcal{P}^{(0)}(k)$

Initialize $r \leftarrow 0$, $c_{\rm st} \leftarrow 0$, and $\Theta_\ell \leftarrow \emptyset,\ \forall \ell \in \{1, 2, \ldots, p\}$

while $c_{\rm st} < 2p - 1$ **do**

  Sort the indices of the subsystems based on the descending order of their local cost in $\mathcal{V}_{\rm so}$

  for $h = 1$ **to** p **do**

    $\ell \leftarrow \mathcal{V}_{\rm so}[1]$

    $\mathcal{V}_{\rm so} \leftarrow \mathcal{V}_{\rm so} \backslash \{\mathcal{V}_{\rm so}[1]\}$

    Compute $\hat{\mathcal{V}}_\ell$, i.e., $\hat{\mathcal{V}}_\ell = \{i : (i,j) \in \mathcal{E}(k), i \in \mathcal{V}_\ell(k),\ j \in \mathcal{V} \backslash \mathcal{V}_\ell(k)\} \backslash \Theta_\ell$

    Compute $f_{\rm c}^{(r)}(\mathcal{V}_\ell(k))$ and solve (5)

    if $f_{\rm c}^{(r)}(\mathcal{V}_\ell(k)) \geq f_{\rm c}(\mathcal{V}_\ell(k) \backslash \{\theta_\ell\})$ **then**

      $\theta_\ell^{(r)}$ is selected

      $c_{\rm st} \leftarrow 0$

      Compute $\Delta f_{{\rm c},\ell}^{(r)}(k)$ according to (6)

      Compute $\Delta f_{{\rm ct},j}^{(r)}(k)$ according to (7) for all $j \in \mathcal{N}_{\mathcal{P},\ell}(k)$

      Decide the subsystem $j^\star$ that will receive θ_ℓ according to (8)

      if $\Delta f_{{\rm ct},j^\star}^{(r)}(k) > 0$ **then**

        $\Theta_\ell \leftarrow \Theta_\ell \cup \{\theta_\ell\}$

      else

        $\Theta_j \leftarrow \emptyset, \quad \forall j \in \mathcal{N}_{\mathcal{P},\ell}(k) \cup \{\ell\}$

        Move θ_ℓ from subsystem ℓ to subsystem $j^\star$, i.e., the partition is updated according to (9)-(10)

      end if

    else

      $c_{\rm st} \leftarrow c_{\rm st} + 1$

    end if

    $r \leftarrow r + 1$

  end for

end while

Output: $\mathcal{P}(k)$

Remark 2. *The number of iterations can be upper bounded by a constant $r_{\max}$ defined by the user. This condition is useful when the available time to compute the partition is limited.* ♢

Remark 3. *The main design parameters in the partitioning approach are α_1 and α_2, which determine the trade-off between two partitioning objectives: 1) minimum coupling among subsystems and 2) balanced number of components per subsystem. Therefore, since these parameters affect the outcome of the partitioning process, i.e., the structure of the subsystems (A and B matrices of the subsystems), they shape the block of the state-feedback gain $\boldsymbol{K}$, which determines the performance of the controlled system. Since a decentralized control scheme takes advantage from minimal coupling, it is better to prioritize objective 1 such that stabilizing controllers can be achieved. This can be done by setting $\alpha_1 \gg \alpha_2$. However, one might also want to set $\alpha_2 \gg \alpha_1$ as long as stabilizing controllers can still be designed so that the communication and computational burden are evenly distributed across subsystems.*

In this paper, the proposed partitioning method is applied along with a decentralized state feedback control scheme, which will be explained in Section 4. However, the application of the partitioning approach is not limited only to this type of controllers and can also be applied to design other types of decentralized or distributed controllers, e.g., Model Predictive Control [35, 36]. Furthermore, different partitioning objectives, depending on the system or control requirements, can also be considered by substituting the cost function (4) with the desired cost function.

4. Decentralized State Feedback Control

4.1. Decentralized State Feedback Control Scheme

Consider now a set of all partitions that have been computed until time instant k using the partitioning approach described in Section 3, i.e., $\overline{\mathcal{P}} = \{\mathcal{P}(\kappa), \kappa = 0, \ldots, k\}$ and consider a given partition $\mathcal{I}$. Note that here we drop the time dependency of the matrices on k, since we are considering a single partition $\mathcal{I}$. We indicate the dynamics of subsystem i of partition $\mathcal{I}$ using the expression

$$x_{\mathcal{I},i}(k+1) = A_{\mathcal{I},ii}x_{\mathcal{I},i}(k) + B_{\mathcal{I},ii}u_{\mathcal{I},i}(k) + \sum_{j\in\mathcal{N}_{\mathcal{I},i}} \left(A_{\mathcal{I},ij}x_{\mathcal{I},j}(k) + B_{\mathcal{I},ij}u_{\mathcal{I},j}(k)\right), \tag{12}$$

where $x_{\mathcal{I},i} \in \mathbb{R}^{n_{\mathcal{I},i}}$ denotes the state vector of partition $\mathcal{I}$, subsystem i, and the matrices $A_{\mathcal{I},ij} \in \mathbb{R}^{n_{\mathcal{I},i}\times n_{\mathcal{I},j}}$, $B_{\mathcal{I},ij} \in \mathbb{R}^{n_{\mathcal{I},ij}\times m_{\mathcal{I},j}}$, for all $j \in \mathcal{N}_{\mathcal{I},i} \bigcup \{i\}$, are respectively the state matrices and the input matrices of the dynamics of partition $\mathcal{I}$, subsystem i. The state vectors $x_{\mathcal{I},j} \in \mathbb{R}^{n_{\mathcal{I},j}}$ with $j \in \mathcal{N}_{\mathcal{I},i}$ are the state vectors of the neighbors of subsystem i in partition $\mathcal{I}$. Similarly, the inputs $u_{\mathcal{I},i} \in \mathbb{R}^{m_{\mathcal{I},i}}$ are the inputs to subsystem i of partition $\mathcal{I}$, while $u_{\mathcal{I},j} \in \mathbb{R}^{m_{\mathcal{I},j}}$ with $j \in \mathcal{N}_{\mathcal{I},i}$ are the inputs of the neighbors of subsystem i in partition $\mathcal{I}$.

It is possible to describe the dynamics of the centralized system of a single partition $\mathcal{I}$ as

$$\boldsymbol{x}_{\mathcal{I}}(k+1) = \boldsymbol{A}_{\mathcal{I}}\boldsymbol{x}_{\mathcal{I}}(k) + \boldsymbol{B}_{\mathcal{I}}\boldsymbol{u}_{\mathcal{I}}(k), \tag{13}$$

where the state vector is $\boldsymbol{x}_{\mathcal{I}} = [x_{\mathcal{I},1}^\top \cdots x_{\mathcal{I},p}^\top]^\top \in \mathbb{R}^n$, the input vector is $\boldsymbol{u}_{\mathcal{I}} = [u_{\mathcal{I},1}^\top \cdots u_{\mathcal{I},p}^\top]^\top \in \mathbb{R}^m$, the state matrix is $\boldsymbol{A}_{\mathcal{I}} = [A_{\mathcal{I},ij}]_{(i,j)\in\{1,\cdots,p\}\times\{1,\cdots,p\}} \in \mathbb{R}^{n\times n}$, and the input matrix is $\boldsymbol{B}_{\mathcal{I}} = [B_{\mathcal{I},ij}]_{(i,j)\in\{1,\cdots,p\}\times\{1,\cdots,p\}} \in \mathbb{R}^{n\times m}$.

Let us now apply to (12) a decentralized static state feedback scheme. For each input $u_{\mathcal{I},i}$, we apply the control law $u_{\mathcal{I},i} = -K_{\mathcal{I},i}x_{\mathcal{I},i}$, where $K_{\mathcal{I},i} \in \mathbb{R}^{m_{\mathcal{I},i}\times n_{\mathcal{I},i}}$ is a gain matrix. We can then combine (13) with the decentralized state feedback control law and obtain

$$\boldsymbol{x}_{\mathcal{I}}(k+1) = (\boldsymbol{A}_{\mathcal{I}} - \boldsymbol{B}_{\mathcal{I}}\boldsymbol{K}_{\mathcal{I}})\boldsymbol{x}_{\mathcal{I}}(k), \tag{14}$$

where $\boldsymbol{K}_{\mathcal{I}} = \mathrm{diag}(K_{\mathcal{I},1}, \ldots, K_{\mathcal{I},p})$ is the centralized gain matrix of partition $\mathcal{I}$. The decentralized gain matrices $K_{\mathcal{I},i}$, $\forall i \in \{1, \ldots, p\}$, are designed such that for each partition the closed-loop centralized matrices $(\boldsymbol{A}_{\mathcal{I}} - \boldsymbol{B}_{\mathcal{I}}\boldsymbol{K}_{\mathcal{I}})$, $\forall \mathcal{I} \in \overline{\mathcal{P}}$, are Schur stable.

Although the closed-loop matrix of each partition, i.e. $\boldsymbol{A}_{\mathcal{I}} - \boldsymbol{B}_{\mathcal{I}}\boldsymbol{K}_{\mathcal{I}}$, is Schur stable, due to the switching nature of the system under consideration, stability of the overall system is not guaranteed. Indeed, as explained in Section 1, stability might arise from the switching between two different asymptotically stable modes. Thus, the continuous change in time of the system dynamics and its partition must be taken directly in account in the stability analysis. In Section 5, we provide a stability analysis using concepts from switching systems theory and directly taking into account the switchings between different dynamics. Before doing that, we make the following assumption:

Assumption 1. *For every partition $\mathcal{I} \in \overline{\mathcal{P}}$, it is possible to obtain a state feedback gain matrix $\boldsymbol{K}_{\mathcal{I}}$ such that the closed-loop system (14) is Schur stable.* ♢

Assumption 1 is needed to prove Proposition 2 but it is not a limiting assumption, since there exist many methods that can provide the matrices $\boldsymbol{K}_{\mathcal{I}}$, as explained in the next subsection.

4.2. Computational Issues

The gain matrices $\boldsymbol{K}_{\mathcal{I}}$, $\forall \mathcal{I} \in \overline{\mathcal{P}}$, can be computed in a centralized fashion with the LMI method proposed in [37, 38] (see Appendix B). With this method, we can obtain $\boldsymbol{K}_{\mathcal{I}}$ matrices that stabilize both the centralized system and the single decentralized subsystems. Moreover, the problem can be solved efficiently using an LMI solver.

When the size of the system is large, a centralized solution could be computationally inefficient. In this case, we have to look for other approaches that can compute the gain matrices $\boldsymbol{K}_{\mathcal{I}}$ in a non-centralized fashion. As explained in [39], one idea could be to apply the concepts from [11], adapted to the discrete-time case. With this strategy, we first stabilize each subsystem separately and then we check the stability of the centralized system in a distributed fashion using a small-gain like condition. As one would expect and as it is highlighted in [11], this method works better when the subsystems are weakly coupled.

5. Stability of the Time-Varying Partition Scheme

In this section, we study the stability of the proposed time-varying partitioning scheme. Before proceeding with the proposition, we present some properties, definitions, and assumptions needed to illustrate the stability of our system.

5.1. Preliminaries

Suppose now that the system dynamics (2) are controlled with a decentralized state feedback strategy, as explained in Section 4. Here, we would like to represent the evolution of the centralized system with a single expression. However, note that the state components in each partition might be grouped differently. In other words, when we consider different partitions, the state components in the centralized state vectors will not be in the same order, in general. To make this concept clearer, let us consider a simple example.

Example 2. *Consider a 5-state component system partitioned into two subsystems, one of them with two state components and the other one with three. Suppose also that the B matrices are all null. There are two partitions and they are denoted by $\mathcal{I}$ and $\mathcal{J}$. Then, let us define the centralized state vector as $\boldsymbol{x} = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 \end{bmatrix}^\top \in \mathbb{R}^5$, where x_1, x_2, x_3, x_4, and x_5 are the state components, and the partitions are*

$$\mathcal{I}: \begin{cases} x_{\mathcal{I},1}(k+1) = A_{\mathcal{I},1} x_{\mathcal{I},1}(k), \\ x_{\mathcal{I},2}(k+1) = A_{\mathcal{I},2} x_{\mathcal{I},2}(k), \end{cases}$$

$$\mathcal{J}: \begin{cases} x_{\mathcal{J},1}(k+1) = A_{\mathcal{J},1} x_{\mathcal{J},1}(k), \\ x_{\mathcal{J},2}(k+1) = A_{\mathcal{J},2} x_{\mathcal{J},2}(k), \end{cases}$$

where $x_{\mathcal{I},1} = \begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix}^\top$, $x_{\mathcal{I},2} = \begin{bmatrix} x_4 & x_5 \end{bmatrix}^\top$, $x_{\mathcal{J},1} = \begin{bmatrix} x_1 & x_3 & x_5 \end{bmatrix}^\top$, $x_{\mathcal{J},2} = \begin{bmatrix} x_2 & x_4 \end{bmatrix}^\top$. Then, $\boldsymbol{x}_\mathcal{I} = \begin{bmatrix} x_{\mathcal{I},1}^\top & x_{\mathcal{I},2}^\top \end{bmatrix}^\top = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 \end{bmatrix}^\top$ and $\boldsymbol{x}_\mathcal{J} = \begin{bmatrix} x_{\mathcal{J},1}^\top & x_{\mathcal{J},2}^\top \end{bmatrix}^\top = \begin{bmatrix} x_1 & x_3 & x_5 & x_2 & x_4 \end{bmatrix}^\top$. As can be seen, vectors $\boldsymbol{x}_\mathcal{I}$ and $\boldsymbol{x}_\mathcal{J}$ have the same components, but they are ordered differently. ♢

Although the centralized state vectors - and thus the closed-loop matrices - are ordered differently, we can still express the equations of the centralized system in a compact form. We can indeed obtain the same order in the state components by applying a permutation transformation with matrices $T_\mathcal{I}$ to the closed-loop matrices $\boldsymbol{A}_\mathcal{I} - \boldsymbol{B}_\mathcal{I}\boldsymbol{K}_\mathcal{I}$, $\forall \mathcal{I} \in \overline{\mathcal{P}}$. After this transformation is applied, we obtain a common order of the state components for all the different partitions. The closed-loop matrix after the transformation would then be $\overline{\boldsymbol{A}}_\mathcal{I} = T_\mathcal{I}^{-1}(\boldsymbol{A}_\mathcal{I} - \boldsymbol{B}_\mathcal{I}\boldsymbol{K}_\mathcal{I})T_\mathcal{I}$. Note that permutation transformations applied to a matrix do not change its eigenvalues. This means that if $\boldsymbol{A}_\mathcal{I} - \boldsymbol{B}_\mathcal{I}\boldsymbol{K}_\mathcal{I}$ is Schur, then $\overline{\boldsymbol{A}}_\mathcal{I}$ is Schur too. We can then finally express the centralized dynamics as

$$\boldsymbol{x}(k+1) = \overline{\boldsymbol{A}}(k)\boldsymbol{x}(k), \tag{15}$$

where $\overline{\boldsymbol{A}}(k)$ is the $\overline{\boldsymbol{A}}_\mathcal{I}$ matrix and partition $\mathcal{I}$ is the active partition at time step k, i.e., $\mathcal{I} = \mathcal{P}(k)$.

We can now present the proof on exponential stability of the time-varying partitioning scheme of Section 3, which is given in the next subsection. Before proceeding with the proof, let us introduce some useful properties, assumptions, and definitions.

Property 1 ([40, 41, Th. 3.5])**.** *If A is a Schur matrix, then $\exists h \in \mathbb{R}_{\geq 1}$, $\exists \lambda_1 \in (0,1)$:, $\forall k \in \mathbb{N}$, $\|A^k\|_2 \leq h\lambda_1^k$.*

Property 2. *Suppose x_{sub} is a sub-vector of $\boldsymbol{x}$, so that $\dim(x_{\text{sub}}) < \dim(\boldsymbol{x})$. Then, $\|x_{\text{sub}}\|_2 \leq \|\boldsymbol{x}\|_2$ since components of x_{sub} are included in $\boldsymbol{x}$. Furthermore, if $\boldsymbol{x}$ and x_{sub} are finite and such that when $\|x_{\text{sub}}\|_2 = 0$ it also holds that $\|\boldsymbol{x}\|_2 = 0$, then it is always possible to find a scalar $d \in \mathbb{R}_{>0}$ such that $\|\boldsymbol{x}\|_2 \leq d\|x_{\text{sub}}\|_2$.*

Proof. It is enough to set $d = \frac{\|\boldsymbol{x}\|_2}{\|x_{\text{sub}}\|_2}$ and the inequality holds. If both $\|\boldsymbol{x}\|_2$ and $\|x_{\text{sub}}\|_2$ are 0, then any $d \in \mathbb{R}_{>0}$ will let the inequality hold. □

Definition 2 ([23, 24])**.** *Consider the discrete-time switching scheme* (15)*. Recall $\sigma(k)$ from Definition 1. Each time there is a* switch*, the dynamics of the system change. Let the number of mode switches between the time steps 0 and k be denoted by $N_\sigma(0,k)$. We define $\tau_{\text{a}} \in \mathbb{R}_{>0}$ as a constant called average dwell time for all the signals $\sigma(k)$ if, for a given $N_0 \geq 0$,*

$$N_\sigma(0,k) \leq N_0 + \frac{k}{\tau_{\text{a}}}. \tag{16}$$

Definition 3. *We denote with λ_{M} the minimum constant such that Property 1 holds for all matrices $\overline{\boldsymbol{A}}_{\mathcal{I}}$, $\forall \mathcal{I} \in \overline{\mathcal{P}}$. Moreover, $h_{\max}$ is the maximum of all h constants in Property 1 for matrices $\overline{\boldsymbol{A}}_{\mathcal{I}}$, $\forall \mathcal{I} \in \overline{\mathcal{P}}$, i.e., if $\|\overline{\boldsymbol{A}}_{\mathcal{I}}^k\|_2 \leq h_{\mathcal{I}}\lambda_{\text{M}}^k$, $\forall \mathcal{I} \in \overline{\mathcal{P}}$ and $k \in \mathbb{N}$, then $h_{\max} = \max_{\mathcal{I} \in \overline{\mathcal{P}}} h_{\mathcal{I}}$.*

Assumption 2. *The average mode dwell time τ_{a}^* of the scheme* (15) *is lower bounded, i.e., $\tau_{\text{a}}^* > -\frac{\log h_{\max}}{\log \lambda_{\text{M}}}$.*

Assumption 3. *The initial state of each subsystem is such that $\|x_{\mathcal{I},i}(0)\|_2 \neq 0$, $\forall i \in \{1, \ldots, p\}$, $\forall \mathcal{I} \in \overline{\mathcal{P}}$.*

Remark 4. *In the framework of slow switching, it is reasonable to suppose that the average dwell time τ_{a}^* should be higher than a certain lower bound [20, 23, 24, 42].*

Remark 5. *As explained later in Remark 7, Assumption 3 is needed to prove Proposition 2. Assumption 3 can be easily checked by looking at the initial values of the state vectors of every subsystem. Moreover, it is still reasonable to assume that, at the initial time instant, none of the state vectors is already at the origin. Nevertheless, in Remark 7 we analyze the case in which Assumption 3 is not satisfied.*

5.2. Stability Analysis

Here, the stability of system (15) is presented in the following proposition.

Proposition 2. *Assume that system* (15) *follows scheme (16) in which N_0 is given, and that Assumptions 1-3 hold. Then, the time-varying partitioned system* (15) *is globally exponentially stable for any average dwell time $\tau_{\text{a}} \geq \tau_{\text{a}}^*$.*

Proof. We follow the proof from [23, 24] using matrices $\overline{\boldsymbol{A}}_{\mathcal{I}}$, $\forall \mathcal{I} \in \overline{\mathcal{P}}$, that we know to be Schur by Assumption 1. In particular, we define k_i, $i = 1, 2, \ldots$ as the time steps at which a mode change occurs, i.e., $k_0 = 0 < \ldots < k_i < k_{i+1} < \ldots$. Let $\boldsymbol{A}_i$ be the matrix $\overline{\boldsymbol{A}}(k)$ in (15) for k between the time steps k_{i-1} and k_i. Then, for any k satisfying $k_i \leq k < k_{i+1}$, we can write the overall state vector $\boldsymbol{x}(k)$ at time step k as

$$\boldsymbol{x}(k) = \boldsymbol{A}_{i+1}^{k-k_i} \boldsymbol{A}_i^{k_i-k_{i-1}} ... \boldsymbol{A}_1^{k_1} \boldsymbol{x}(0).$$

Applying to both sides the norm operator, we get

$$\|\boldsymbol{x}(k)\|_2 \leq \|\boldsymbol{A}_{i+1}^{k-k_i}\|_2 \cdot \|\boldsymbol{A}_i^{k_i-k_{i-1}}\|_2 ... \|\boldsymbol{A}_1^{k_1}\|_2 \cdot \|\boldsymbol{x}(0)\|_2. \tag{17}$$

Consider now Property 1 and let us apply it to all the matrices $\boldsymbol{A}_i$ in (17). Let h_i be the constant h in Property 1 associated to $\boldsymbol{A}_i$ and let $\lambda_{\mathrm{M}} \in (0, 1)$ be the constant defined in Definition 3. We can then replace every matrix norm $\|\boldsymbol{A}_i\|$ in (17) by $h_i \lambda_{\mathrm{M}}^{k_i-k_{i-1}}$, yielding

$$\|\boldsymbol{x}(k)\|_2 \leq h_{i+1} h_i ... h_1 \lambda_{\mathrm{M}}^k \|\boldsymbol{x}(0)\|_2 = \left(\prod_{j=1}^{i+1} h_j \right) \lambda_{\mathrm{M}}^k \|\boldsymbol{x}(0)\|_2.$$

Let $h_{\max}$ be as in Definition 3. Then,

$$\|\boldsymbol{x}(k)\|_2 \leq \left(\prod_{j=1}^{i+1} h_j \right) \lambda_{\mathrm{M}}^k \|\boldsymbol{x}(0)\|_2 \leq c h_{\max}^{N_\sigma(0,k)} \lambda_{\mathrm{M}}^k \|\boldsymbol{x}(0)\|_2, \tag{18}$$

where $c = h_{\max}$ and $N_\sigma(0, k)$ is defined in Definition 2. Since $h_{\max} \geq 1$, we define $h_0 = h_{\max}^{N_0}$ and from Assumption 2 we can define a $\lambda \in (\lambda_{\mathrm{M}}, 1)$ such that $\lambda = \lambda_{\mathrm{M}} h_{\max}^{\frac{1}{\tau_{\mathrm{a}}}}$. From (16) we then obtain

$$h_{\max}^{N_\sigma(0,k)} \leq h_{\max}^{N_0} h_{\max}^{\frac{k}{\tau_{\mathrm{a}}}} = h_0 h_{\max}^{\frac{k}{\tau_{\mathrm{a}}}} = h_0 \left(\frac{\lambda}{\lambda_{\mathrm{M}}} \right)^k. \tag{19}$$

We apply (19) to (18) to get

$$\|\boldsymbol{x}(k)\|_2 \leq c h_0 \lambda^k \|\boldsymbol{x}(0)\|_2. \tag{20}$$

Lastly, we apply Assumption 3 and Property 2 and get

$$\|x_{\mathcal{I},i}(k)\|_2 \leq \|\boldsymbol{x}(k)\|_2 \leq c h_0 \lambda^k \|\boldsymbol{x}(0)\|_2 \leq c d h_0 \lambda^k \|x_{\mathcal{I},i}(0)\|_2, \tag{21}$$

$\forall i \in \{1, \ldots, p\}$, $\forall \mathcal{I} \in \overline{\mathcal{P}}$. Thus, each subsystem in $\overline{\mathcal{P}}$ is globally exponentially stable for any average dwell time $\tau_{\mathrm{a}} \geq \tau_{\mathrm{a}}^*$. Since (21) holds for each possible subsystem in each partition of $\overline{\mathcal{P}}$, it holds simultaneously for all the subsystems in all the partitions. Therefore, each subsystem of the closed-loop system is exponentially stable. □

Remark 6. *Although the first part of the proof of Proposition 2 has already appeared in [23, 24], here we add the extra step* (21) *in order to adapt the proof to our case since [23, 24] do not consider a decentralized control scheme.*

Remark 7. *Without Assumption 3, we cannot apply the second part of Property 2 to* (20)*. However, we can still guarantee (simple) exponential stability, since for* (21) *we can write* $\|\boldsymbol{x}(k)\|_2 \leq c\lambda^k \|\boldsymbol{x}(0)\|_2 \leq c\lambda \|\boldsymbol{x}(0)\|_2$ *and choosing* $c_1 = c\lambda\|\boldsymbol{x}(0)\|_2$ *we get* $\|x_{\mathcal{I},i}\|_2 \leq \|\boldsymbol{x}(k)\|_2 \leq c_1$, $\forall i \in \{1,\ldots,p\}$, $\forall \mathcal{I} \in \overline{\mathcal{P}}$*. Then, the state* $x_{\mathcal{I},i}$ *is simply stable as in [43] with* c_1 *as specified before and* $\forall c_2 > 0$.

Remark 8. *According to Proposition 2, stability is not guaranteed if the time-varying partitioning scheme switches faster than* τ_{a}^**. This result is related to the* slow *switching systems, in which stability is guaranteed only if the average dwell time is higher than a certain lower bound [20].*

6. Overall Scheme

In this section we explain the overall partitioning and control scheme of our approach. Moreover, we also explain how to update the average dwell time τ_{a}^* that guarantees exponential stability in view of Proposition 2.

6.1. Controller Structure

The overall control scheme is shown in Figure 2. The controller has a hierarchical structure and it is divided in two different parts: a centralized one, the *coordinator*, and decentralized controllers. The tasks of the coordinator are to detect changes in the system, to update the partition, and to compute the decentralized state-feedback gains and the bound of the average dwell time. These updates are only carried out when a mode switch is detected. Once the updates have been communicated to the decentralized controllers and subsystems have been created, then there is no further task for the coordinator, except for checking for changes in the system at each time step. Indeed, if there is no change in either the mode or the partition of the controlled system, then the system can be controlled only by the decentralized state feedback controller, which can stabilize the overall system in view of the result of Proposition 2. In addition, it is assumed that the time required for the coordinator to perform its tasks is smaller than the sampling time. It implies that the partition, state feedback gains, and the bound of the average dwell time are updated at the same time instant as the one at which the switch is detected. In practice, the satisfaction of this assumption depends on the computational power of the coordinator and the instrumentation of the system.

6.2. Updating Scheme

For the purpose of updating the partition and the decentralized state feedback gains, the coordinator records the modes and their corresponding partition and state feedback gains in a library. The advantage of this approach is to speed up the updating process. This approach is particularly effective for systems that have periodical behavior, i.e., the modes of the system appear periodically. Therefore, when the coordinator detects a switch, it firstly checks its library. If the current mode has been recorded, it can immediately provide a suitable partition and stabilizing state feedback gains to the decentralized controllers. Otherwise, it will perform the partitioning method proposed in Section 3. Afterwards, the coordinator will compute

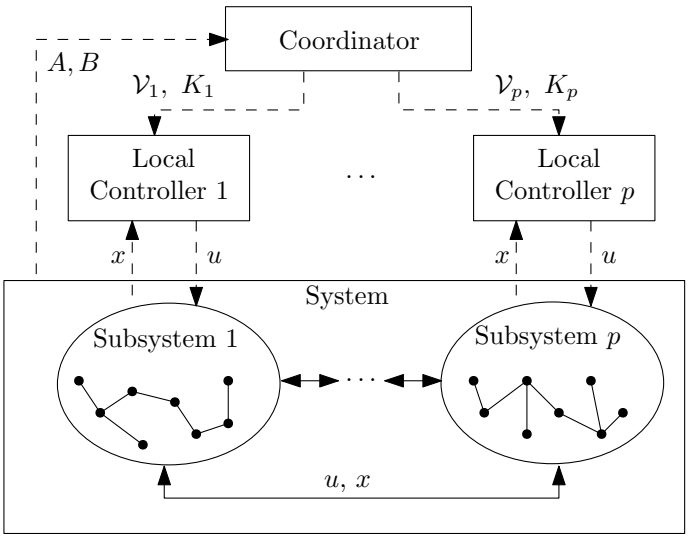


Figure 2: Overall control scheme, with a centralized coordinator, the system, and its subsystems. The dashed arrows represent the exchange of information. The solid arrows represent the coupling between subsystems.

the new decentralized gain matrix. The new mode, partition, and gain matrix will then be recorded in the library.

If a new mode appears in the system, we have to update the value of τ_{a}^*, because it depends on the characteristics of the current matrices $\bar{\boldsymbol{A}}_{\mathcal{I}}$. Moreover, the update of the value of τ_{a}^* is needed since, as explained in Section 2, the modes have a recurrent behavior, i.e., they might be active again after some time during which other modes were active. Therefore, when updating the value of τ_{a}^*, we need to consider the inactive modes. We then present the following procedure for updating τ_{a}^*:

1. For each new mode, we consider the closed-loop matrix $\bar{\boldsymbol{A}}_{\mathcal{I}}$ and we compute the h and λ_1 constants associated to this matrix as in Property 1;
2. We check whether $h > h_{\max}$ and in case this is true, we update $h_{\max}$ as $h_{\max} = h$. We also check whether $\lambda_1 > \lambda_{\mathrm{M}}$ and if the answer is true, we update λ_{M} as $\lambda_{\mathrm{M}} = \lambda_1$. If no updates are carried out at this step, we skip the next step;
3. We update the value of τ_{a}^* as
$$\tau_{\mathrm{a}}^* = \frac{\ln(h_{\max})}{\ln(1-\varepsilon) - \ln(\lambda_{\mathrm{M}})}, \tag{22}$$
where ε is a small positive constant.

Remark 9. *Note that in Step 3 of the above procedure, we have updated τ_{a}^* following Assumption 2 and maximizing the value of λ, which is set to $1-\varepsilon$. By doing so, we choose the least conservative value for τ_{a}^* while still satisfying Assumption 2.*

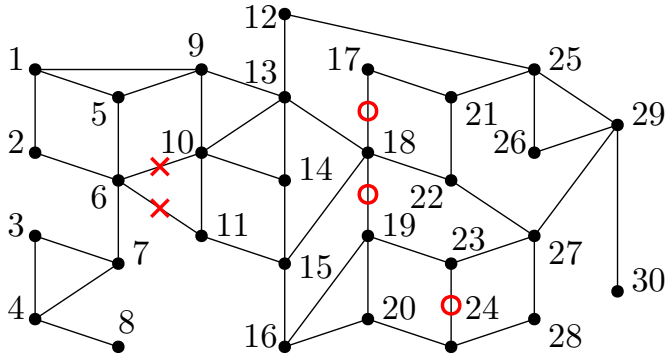


Figure 3: The network topology considered. Vertices represent areas and edges represent couplings between the areas. In mode 1, all links are working correctly, while in the other two modes some links are broken. A red '×' marker indicates the broken links in mode 2, while a red '◯' marker indicates the broken links in mode 3.

7. Case Study

We consider the problem of controlling a power network consisting of 30 areas (vertices) that are connected by tie-lines as shown in Figure 3 through automatic generation control, inspired by [44]. Each area has a thermal turbine as the generator and load perturbations. Since the areas are interconnected, electrical power may flow between them. The control objective is to stabilize the frequency and tie-line power deviation of each area. For modeling each area, we follow the dynamical model provided in [45]. The state vector of area $i \in \mathcal{V}_x$ is $x_i^\top = \begin{bmatrix}\Delta f_i & \Delta P_{\mathrm{G}_i} & \Delta P_{\mathrm{R}_i} & X_{\mathrm{E}_i} & \Delta P_{\mathrm{ref}_i} & \Delta P_{\mathrm{tie}_i}\end{bmatrix}$, where the state components represent respectively the deviation with respect to the nominal value in frequency, turbine output power, mechanical power during steam reheat, governor valve position, variable achieving integral control, and the tie-line power. Furthermore, dynamical coupling is present in the model, since the dynamics of the states of one area are influenced by the states of some of the neighboring areas. The state space matrices used in the numerical simulations are provided in the appendix.

In this network, there are some vulnerable links that might be temporarily broken during the operation of the system. Since these faults may arise in the system, the system has switching behavior. Moreover, since we assume the faults do not happen so fastly, we are in the framework of *slow* switching systems. We design a decentralized state feedback controller for this network as explained in Section 4.

In our simulation, we consider 3 different modes. Mode 1 represents the normal operation, while the other modes represent the network with physical faults, i.e., broken links as shown in Figure 3, but we consider that nodes can still communicate with each other. Furthermore, we assume that we have five decentralized controllers, implying that the system must be partitioned into five subsystems. The scenario of the simulation is as follows: For $k \leq 26$, the system operates normally, i.e., in mode 1. At $k = 27$, the system switches to mode 2, and at $k = 134$, the system switches to mode 3. Following the proposed approach, the system is partitioned with the algorithm proposed in Section 3 and stabilizing decentralized state-feedback gains are computed by using the method proposed in [37] at $k = 0, 27, 134$, when the system switches its mode. The

partitioning results of all modes are shown in Figure 4. With the three different topologies, the partitioning algorithm produces also three different partitioning topologies; thus the partition is adapted to the mode.

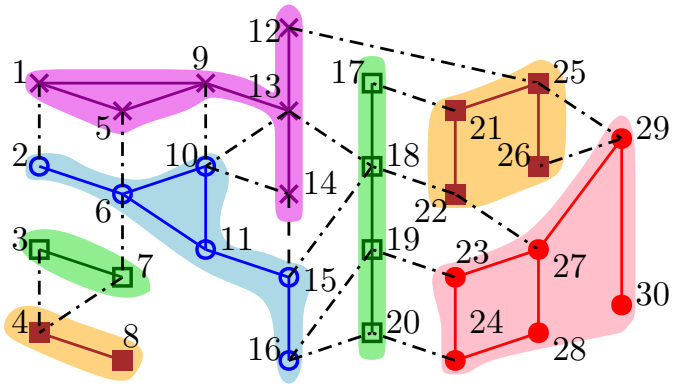
In this simulation, the sampling time is $T = 1$ s and the closed-loop system is simulated for 200 time steps. The initial state of each area $i \in \mathcal{V}_x$ is $x_i(0) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^\top$, which represents an impulse disturbance in the frequency deviation. The value of τ_{a}^* for the first mode and the first partition is 23.76 and then it is updated to 24.05 at $k = 27$ and finally it is updated again to 26.63 at $k = 124$. Since the average dwell time of the system is 80.5, which is larger than τ_{a}^*, the stability of the system is guaranteed.

The frequency deviation Δf and the power ΔP_{tie} at some areas are shown in Figure 5. It is possible to observe that the time-varying partitioned closed-loop system is asymptotically stable, even after the system switches to other modes, as expected from Proposition 2. Moreover, Figure 6 shows that the system becomes unstable if the partition and the decentralized state-feedback gains are not switched after the mode changes, i.e., the system switches to different modes but the partition and the gain matrices of mode 1 are kept for the whole simulation. This behavior was also presented in Example 1. It is also interesting to notice in Figure 6, between time steps 27 and 134, that, although partitioning topologies 1 and 2 do not differ too much, instability arises if the partitioning topology of mode 1 is applied to mode 2. This is therefore a further motivation for the time-varying partitioning approach proposed in this article. Note that in this case study, we have compared the online partitioning approach with the static one since, to the best of our knowledge, there is no other online system partitioning algorithm proposed yet.

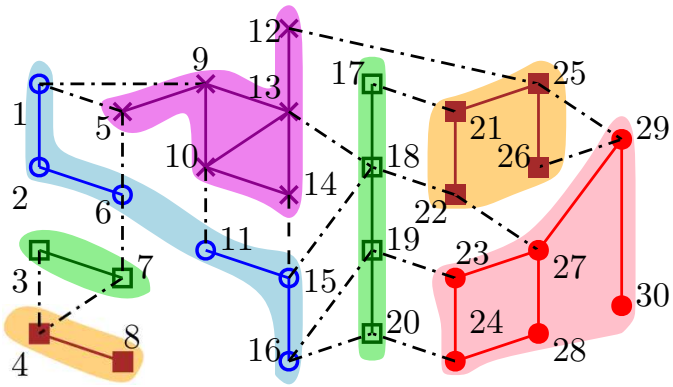
8. Conclusions

This article has presented a novel multi-step graph-based partitioning method for a class of large-scale linear switching systems. The proposed algorithm is computationally inexpensive and can be applied online when a change in the system occurs. Moreover, a decentralized state-feedback controller is applied to the obtained subsystems in order to stabilize the subsystem. A further analysis of the closed-loop system, modeled as an autonomous switching system, is provided. The stability of the system is guaranteed if the average dwell time of the system is larger than a lower bound. Additionally, a case study of automatic generation control of multi-area power network, in which some link failures might happen, has been discussed. The results show that the proposed method can stabilize the closed-loop system while, if not applied, the system might become unstable.

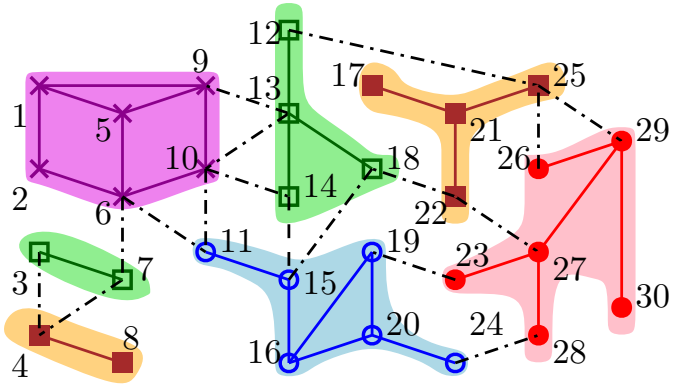
As future work, the proposed method can be extended by considering a *dynamic* partitioning, i.e. a method in which nodes are moved dynamically between subsystems, rather than computing a whole new partition. Moreover, another interesting related work is to apply other types of control algorithms to the system, e.g. distributed or decentralized model predictive control. In that case, different sets of tools are required to show the stability of the closed-loop systems. In addition, we also consider to extend our approach to switching systems in which some of the modes are not stabilizable.



a. Partitioning result of mode 1.



b. Partitioning result of mode 2.



c. Partitioning result of mode 3.

Figure 4: The partitioning results of all modes. Vertices in subsystem 1 are indicated by •, vertices in subsystem 2 are indicated by ∘, vertices in subsystem 3 are indicated by □, vertices in subsystem 4 are indicated by ■, and vertices in subsystem 5 are indicated by ×, while the edges that couple different subsystems are indicated by dashed-dotted lines.

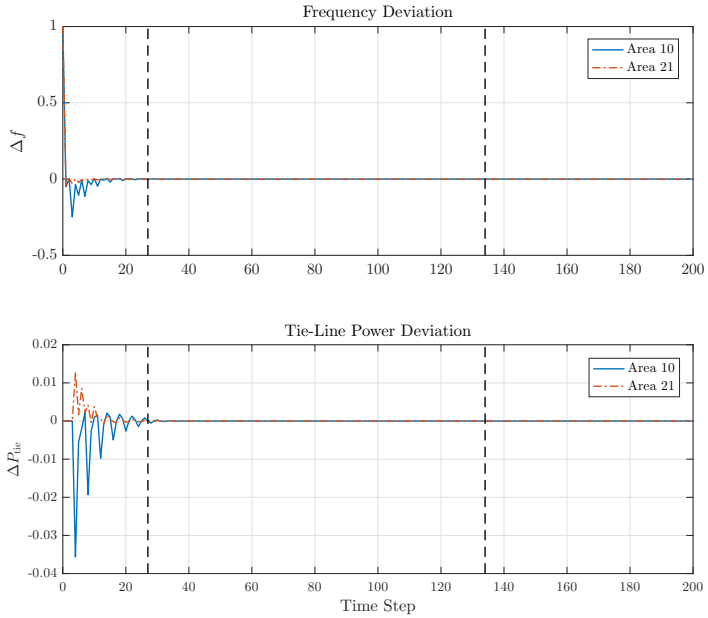


Figure 5: The deviation in frequency and tie-line power of areas 10 and 21 in the case in which the partition is changed together with the modes. The time steps at which the switches occur are indicated by a vertical dashed black line.

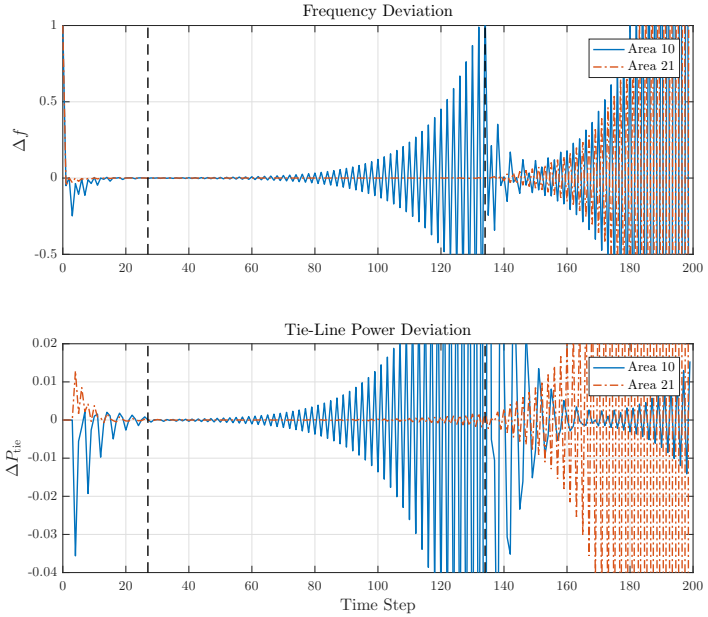


Figure 6: The deviation of frequency and tie-line power of areas 10 and 21 in the case in which both the partitioning topology and the gain matrix related to mode 1 are kept for the whole simulation, even after a mode switch. The time steps at which the switches occur are indicated by a vertical dashed black line.

Appendix A. Parameters of the simulations

The state-space matrices of each area, which are adapted from [44], are as follows:

$$A_{ii} = \begin{bmatrix} -0.05 & 6 & 0 & 0 & 0 & -6 \\ 0 & -0.1 & -1.01 & 1.11 & 0 & 0 \\ 0 & 0 & -3.33 & 3.33 & 0 & 0 \\ -2.08 & 0 & 0 & -5 & 5 & 0 \\ -0.0255 & 0 & 0 & 0 & 0 & -0.06 \\ \gamma_i & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$B_{ii} = \begin{bmatrix} 0 & 0 & 0 & 5 & 0 & 0 \end{bmatrix}^\top$, where $\gamma_i = \sum_{j=1}^{|\mathcal{N}_i|} 0.056$.

Furthermore, for all the couplings $(i,j) \in \{(5,1), (1,2), (4,3), (8,4), (9,5), (2,6), (5,6), (7,6), (3,7), (4,7), (1,9), (13,9), (6,10), (9,10), (6,11), (10,11), (13,12), (10,13), (18,13), (10,14), (13,14), (15,14), (11,15), (15,16), (19,16), (18,17), (15,18), (19,18), (22,18), (23,19), (16,20), (19,20), (17,21), (25,21), (21,22), (27,23), (20,24), (23,24), (12,25), (29,25), (25,26), (29,26), (22,27), (28,27), (24,28), (27,29), (29,30)\}$, the coupling state space matrices are the following:

$$A_{ij} = \begin{bmatrix} 0_{5\times1} & 0_{5\times5} \\ -0.055 & 0_{1\times5} \end{bmatrix}, \; A_{ji} = \begin{bmatrix} 0_{4\times1} & 0_{4\times4} & 0_{4\times1} \\ 0 & 0_{1\times4} & 0.06 \\ -0.055 & 0_{1\times4} & 0 \end{bmatrix}.$$

Appendix B. LMI method for Decentralized State-Feedback Control Design [37, 38]

In order to find the gain matrices $\boldsymbol{K}$ for decentralized state-feedback control of a system that consists of p subsystems, the problem of finding a gain matrix $\boldsymbol{K}$ and a matrix $\boldsymbol{P} \succ 0$, such that R1-3

$$\begin{cases} \boldsymbol{P} \succ 0, \\ (\boldsymbol{A} + \boldsymbol{B}\boldsymbol{K})^\top \boldsymbol{P} (\boldsymbol{A} + \boldsymbol{B}\boldsymbol{K}) - \boldsymbol{P} \prec 0, \end{cases} \tag{B.1}$$

is solved using an LMI procedure. We define two matrices $\boldsymbol{S}$ and $\boldsymbol{Y}$ such that $\boldsymbol{K} = \boldsymbol{Y}\boldsymbol{S}^{-1}$ and $\boldsymbol{S} = \boldsymbol{P}^{-1}$ and we solve with respect to $\boldsymbol{S}$ and $\boldsymbol{Y}$ the following LMI procedure:

$$\begin{bmatrix} \boldsymbol{S} & \boldsymbol{S}\boldsymbol{A}^\top + \boldsymbol{Y}^\top \boldsymbol{B}^\top \\ \boldsymbol{A}\boldsymbol{S} + \boldsymbol{B}\boldsymbol{Y} & \boldsymbol{S} \end{bmatrix} \succ 0, \tag{B.2}$$

$$\begin{bmatrix} \boldsymbol{S}_{ii} & \boldsymbol{S}_{ii}\boldsymbol{A}_{ii}^\top + \boldsymbol{Y}_{ii}^\top \boldsymbol{B}_{ii}^\top \\ \boldsymbol{A}_{ii}\boldsymbol{S}_{ii} + \boldsymbol{B}_{ii}\boldsymbol{Y}_{ii} & \boldsymbol{S}_{ii} \end{bmatrix} \succ 0, \tag{B.3}$$

subject to

$$\boldsymbol{S}_{ij} = 0, \boldsymbol{Y}_{ij} = 0, \; \forall i, j = 1, \ldots, p, \; (i \neq j), \tag{B.4}$$

where, for $i, j = 1, \ldots, p$, $\boldsymbol{S}_{ij} \in \mathbb{R}^{n_i \times n_j}$, $\boldsymbol{Y}_{ij} \in \mathbb{R}^{m_i \times n_j}$ are the block entries of the matrices $\boldsymbol{S}$ and $\boldsymbol{Y}$. The LMI (B.2) guarantees the stability of the centralized system, while (B.3) stabilizes each subsystem. A more detailed procedure can be found in [38].

References

[1] A. Núñez, C. Ocampo-Martinez, J. Maestre, B. De Schutter, Time-varying scheme for noncentralized model predictive control of large-scale systems, Mathematical Problems in Engineering 2015 (2015) 1–17.

[2] C. Ocampo-Martinez, V. Puig, G. Cembrano, J. Quevedo, Application of predictive control strategies to the management of complex networks in the urban water cycle, IEEE Control Systems Magazine 33 (1) (2013) 15–41.

[3] Z. Zhou, B. De Schutter, S. Lin, Y. Xi, Two-level hierarchical model-based predictive control for large-scale urban traffic network, IEEE Transactions on Control Systems Technology 25 (2) (2017) 496–508.

[4] M. Kraning, E. Chu, J. Lavaei, S. Boyd, Dynamic network energy management via proximal message passing, Foundations and Trends in Optimization 1 (2) (2014) 73–126.

[5] X. Xiao, Z. Mao, Decentralized guaranteed cost stabilization of time-delay large-scale systems based on reduced-order observers, Journal of the Franklin Institute 348 (9) (2011) 2689–2700.

[6] M. Guinaldo, J. Sánchez, R. Dormido, S. Dormido, Distributed control for large-scale systems with adaptive event-triggering, Journal of the Franklin Institute 353 (3) (2016) 735–756.

[7] L. Bakule, Decentralized control: An overview, Annual Reviews in Control 32 (1) (2008) 87–98.

[8] N. Sandell, P. Varaiya, M. Athans, M. Safonov, Survey of decentralized control methods for large scale systems, IEEE Transactions on Automatic Control 23 (2) (1978) 108–128.

[9] R. Scattolini, Architectures for distributed and hierarchical model predictive control a review, Journal of Process Control 19 (5) (2009) 723–731.

[10] X. Ge, F. Yang, Q.-L. Han, Distributed networked control systems: A brief overview, Information Sciences 380 (2017) 117–131.

[11] D. D. Šiljak, Decentralized Control of Complex Systems, Academic Press, Inc., 1991.

[12] J. Lunze, Feedback Control of Large Scale Systems, Prentice Hall PTR, Upper Saddle River, NJ, USA, 1992.

[13] M. Sezer, D. Šiljak, On structural decomposition and stabilization of large-scale control systems, IEEE Transactions on Automatic Control 26 (2) (1981) 439–444.

[14] M. Sezer, D. Šiljak, Nested ϵ-decompositions and clustering of complex systems, Automatica 22 (3) (1986) 321–331.

[15] M. Nayeripour, H. Fallahzadeh-Abarghouei, E. Waffenschmidt, S. Hasanvand, Coordinated online voltage management of distributed generation using network partitioning, Electric Power Systems Research 141 (2016) 202–209.

[16] J. Guo, G. Hug, O. K. Tonguz, Intelligent partitioning in distributed optimization of electric power systems, IEEE Transactions on Smart Grid 7 (3) (2016) 1249–1258.

[17] J. Barreiro-Gomez, C. Ocampo-Martinez, N. Quijano, Partitioning for large-scale systems: A sequential distributed MPC design, IFAC-PapersOnLine 50 (1) (2017) 8838–8843.

[18] C. Ocampo-Martinez, S. Bovo, V. Puig, Partitioning approach oriented to the decentralised predictive control of large-scale systems, Journal of Process Control 21 (5) (2011) 775–786.

[19] F. Blanchini, S. Miani, Switching and switched systems, in: Set-Theoretic Methods in Control, Springer International Publishing, 2015, Ch. 9, pp. 405–466.

[20] H. Lin, P. J. Antsaklis, Stability and stabilizability of switched linear systems: A survey of recent results, IEEE Transactions on Automatic Control 54 (2) (2009) 308–322.

[21] D. Liberzon, Switching in Systems and Control, Systems & Control: Foundations & Applications, Birkhäuser Boston, 2003.

[22] R. A. Decarlo, M. S. Branicky, S. Pettersson, B. Lennartson, Perspectives and results on the stability and stabilizability of hybrid systems, Proceedings of the IEEE 88 (7) (2000) 1069–1082.

[23] G. Zhai, B. Hu, K. Yasuda, A. N. Michel, Qualitative analysis of discrete-time switched systems, in: Proceedings of the American Control Conference, Vol. 3, 2002, pp. 1880–1885.

[24] G. Zhai, B. Hu, K. Yasuda, A. N. Michel, Stability and $\mathcal{L}_2$ gain analysis of discrete-time switched systems, Transactions of the Institute of Systems, Control and Information Engineers 15 (3) (2002) 117–125.

[25] W. Xiang, J. Lam, J. Shen, Stability analysis and L1-gain characterization for switched positive systems under dwell-time constraint, Automatica 85 (2017) 1–8.

[26] X. Zhao, Y. Yin, B. Niu, X. Zheng, Stabilization for a class of switched nonlinear systems with novel average dwell time switching by T-S fuzzy modeling, IEEE Transactions on Cybernetics 46 (8) (2016) 1952–1957.

[27] X. Zhao, Y. Yin, X. Zheng, State-dependent switching control of switched positive fractional-order systems, ISA Transactions 62 (2016) 103–108, SI: Control of Renewable Energy Systems.

[28] J. C. Geromel, P. Colaneri, Stability and stabilization of discrete time switched systems, International Journal of Control 79 (7) (2006) 719–728.

[29] A. Kundu, D. Chatterjee, On stability of discrete-time switched systems, Nonlinear Analysis: Hybrid Systems 23 (2017) 191–210.

[30] M. Huang, L. Ma, G. Zhao, X. Wang, Z. Wang, Input-to-state stability of discrete-time switched systems and switching supervisory control, in: 2017 IEEE Conference on Control Technology and Applications (CCTA), 2017, pp. 910–915.

[31] A. Kundu, P. K. Mishra, D. Chatterjee, Stabilizing discrete-time switched systems with inputs, in: 2015 54th IEEE Conference on Decision and Control (CDC), 2015, pp. 4897–4902.

[32] B. W. Kernighan, S. Lin, An efficient heuristic procedure for partitioning graphs, The Bell System Technical Journal 49 (2) (1970) 291–307.

[33] J. R. Gilbert, E. Zmijewski, A parallel graph partitioning algorithm for a message-passing multiprocessor, International Journal of Parallel Programming 16 (6) (1987) 427–449.

[34] G. Karypis, V. Kumar, A fast and high quality multilevel scheme for partitioning irregular graphs, SIAM Journal on Scientific Computing 20 (1) (1998) 359–392.

[35] E. F. Camacho, C. Bordons Alba, Model Predictive Control, Advanced Textbooks in Control and Signal Processing, Springer London, 2013.

[36] J. Maciejowski, Predictive Control with Constraints, Prentice Hall, England., 2002.

[37] G. Betti, M. Farina, R. Scattolini, Realization issues, tuning, and testing of a distributed predictive control algorithm, Journal of Process Control 24 (4) (2014) 424–434.

[38] G. Betti, M. Farina, R. Scattolini, Distributed predictive control for tracking constant references, in: 2012 American Control Conference (ACC), 2012, pp. 6364–6369.

[39] M. Farina, G. Betti, R. Scattolini, Distributed predictive control of continuous-time systems, Systems & Control Letters 74 (Supplement C) (2014) 32–40.

[40] J. LaSalle, The Stability and Control of Discrete Processes, Applied Mathematical Sciences, Springer, 1986.

[41] D. A. Dowler, Bounding the norm of matrix powers, Master's thesis, Brigham Young University-Provo (2013).

[42] J. P. Hespanha, A. S. Morse, Stability of switched systems with average dwell-time, in: Proceedings of the 38th IEEE Conference on Decision and Control, Vol. 3, 1999, pp. 2655–2660.

[43] V. C. Aitken, H. M. Schwartz, On the exponential stability of discrete-time systems with applications in observer design, IEEE Transactions on Automatic Control 39 (9) (1994) 1959–1962.

[44] X.-B. Chen, S. S. Stankovic, Overlapping decentralized approach to automation generation control of multi-area power systems, International Journal of Control 80 (3) (2007) 386–402.

[45] H. Zeynelgil, A. Demiroren, N. Sengor, The application of ANN technique to automatic generation control for multi-area power system, International Journal of Electrical Power & Energy Systems 24 (5) (2002) 345–354.