53rd IEEE Conference on Decision and Control
December 15-17, 2014. Los Angeles, California, USA

Distributed Linear Supervisory Control

Ali Khanafer, Tamer Başar, and Daniel Liberzon

Abstract— **In this work, we propose a distributed version of the logic-based supervisory adaptive control scheme. Given a network of agents whose dynamics contain unknown parameters, the distributed supervisory control scheme is used to assist the agents to converge to a certain set-point without requiring them to have explicit knowledge of that set-point. Unlike the classical supervisory control scheme where a centralized supervisor makes switching decisions among the candidate controllers, in our scheme, each agent is equipped with a local supervisor that switches among the available controllers. The switching decisions made at a certain agent depend only on the information from its neighboring agents. We apply our framework to the distributed averaging problem in the presence of large modeling uncertainty and support our findings by simulations.**

I. Introduction

Logic-based switching supervisory control has been proposed as a method to overcome limitations of adaptive control schemes [1]. A fundamental difference between the two approaches is that while adaptive control requires continuous tuning of parameters, supervisory control relies on logic-based switching among a collection of candidate controllers. Continuous tuning suffers from well-known issues such as loss of stabilizability. In the classical supervisory control scheme, a centralized supervisor estimates the state of the plant, and based on the history of estimation errors, it activates a certain candidate controller. For a more detailed study of supervisory control, see Chapter 6 of [2].

Supervisory control has been used in various problems and applications [3]–[11]. In [3], [4], the set-point control problem has been studied using a supervisory control framework. It has also been utilized in path-following problems for underactuated systems with large modeling uncertainties [6]. Recently, supervisory control has been extended to addresses the problem of stabilizing uncertain systems with quantized outputs [11].

In this work, and motivated by its attractive properties, we extend the supervisory control framework to a distributed setting. A distributed version of supervisory control can have wide applications in stabilization and tracking problems over networked systems in the presence of large modelling uncertainties.

Statement of Contributions

The main contribution of this paper is extending the centralized supervisory control framework to a distributed setting. We first provide a detailed description of the main components in this scheme. We prove that when the set in which the unknown parameters take values is finite, the switching stops in finite time at each node. Further, we provide sufficient conditions for achieving set-point tracking using this framework without requiring the individual agents to have explicit knowledge of the desired set-point. Finally, we apply this scheme to the distributed averaging problem in the presence of unknown parameters, and we support our findings with simulations.

Notation and Terminology

We denote the i-th row of a matrix $X \in \mathbb{R}^{n\times m}$ by $[X]_i \in \mathbb{R}^m$, and the (i,j)-th entry of that matrix by $[X]_{ij} \in \mathbb{R}$. Similarly, we denote the i-th entry of a vector $x \in \mathbb{R}^n$ by $[x]_i \in \mathbb{R}$. We adopt the game-theoretic notation x_{-i} to mean the collection of vectors x_j for all $j \neq i$. The identity matrix is denoted by I, and the all-ones vector is denoted by $\mathbf{1}$.

A directed graph is a pair $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nodes and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the set of edges. Given $\mathcal{G}$, we denote an edge from node $i \in \mathcal{V}$ to node $j \in \mathcal{V}$ by (i,j). When $(i,j) \in \mathcal{E}$ if and only if $(j,i) \in \mathcal{E}$, we call the graph undirected. We call an undirected graph *connected* if it contains a path between any two nodes in $\mathcal{V}$. We use the words "nodes" and "agents" interchangeably.

Organization

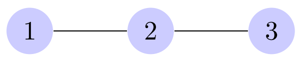
In Section II, we introduce the system model and present the problem formulation. The main components of the distributed supervisory control scheme are provided in Section III. Section IV contains the stability analysis of the proposed scheme. An application to the distributed averaging problem is presented in Section V. We conclude the paper in Section VI and provide ideas for future work.

II. System Model

Consider a network with n nodes, and let $x \in \mathbb{R}^n$ be the state of the network, where $[x]_i \in \mathbb{R}$ is the state of node i. It is possible to extend this setting to the case where the state of the i-th agent is k_i-dimensional, where $k_1 + \ldots + k_n = n$; however, in this paper, we restrict our attention to the case where the state of each node is scalar for simplicity. Let $u \in \mathbb{R}^n$ be a vector consisting of the inputs to all the nodes with $[u]_i \in \mathbb{R}$ being a scalar input to node i. Further, let $y \in \mathbb{R}^n$ be a vector consisting of the outputs of all the nodes with $[y]_i \in \mathbb{R}$ being a scalar output of node i. Similar to the state variables, it is possible to allow the nodes to take multiple inputs and produce multiple outputs, and the restriction to

Research supported in part by an AFOSR MURI Grant FA9550-10-1-0573 and in part by a grant through the Information Trust Institute of the University of Illinois.

The authors are with the Coordinated Science Laboratory, ECE Department, University of Illinois at Urbana-Champaign, USA {khanafe2,basar1,liberzon}@illinois.edu



$$A_p = \begin{bmatrix} 0 & * & 0 \\ * & 0 & * \\ 0 & * & 0 \end{bmatrix}$$

Fig. 1: A path graph with 3 nodes and its corresponding A_p.

the single-input single-output set-up is for purpose of clarity in presentation.

The network is described by a graph whose topology is unknown, i.e., the interconnections among the n nodes are not known. Let $\mathcal{P} = \{1, \ldots, r\}$ be a finite index set. To each $p \in \mathcal{P}$, we associate a graph $\mathcal{G}_p = (\mathcal{V}_p, \mathcal{E}_p)$, where $\mathcal{V}_p$ is the set of vertices, and $\mathcal{E}_p \subseteq \mathcal{V}_p \times \mathcal{V}_p$ is the set of edges. The index $p^\star \in \mathcal{P}$ is unknown to the nodes, and its corresponding graph, $\mathcal{G}_{p^\star}$, describes the actual network under study. The graphs $\mathcal{G}_p$, $p \neq p^\star$, are different possibilities of what $\mathcal{G}_{p^\star}$ might be. To each graph $\mathcal{G}_p$, there corresponds a linear dynamical system represented by a triple (A_p, B_p, C_p), where $A_p, B_p, C_p \in \mathbb{R}^{n\times n}$. Each triple represents a different possibility of the actual system $(A_{p^\star}, B_{p^\star}, C_{p^\star})$ that governs the dynamics of the network. In particular, we assume that the nodes operate according to the following linear dynamics:

$$\begin{aligned} \dot{x} &= A_{p^\star}x + B_{p^\star}u, \quad x(0) = x_0, \\ y &= C_{p^\star}x. \end{aligned} \tag{1}$$

We define the neighborhood of node i in the graph $\mathcal{G}_p$ as

$$N_p(i) = \{j \in \mathcal{V}_p : (j,i) \in \mathcal{E}_p\}.$$

Note that we have not explicitly included i in $N_p(i)$ to allow for applications where node i is not able to measure its own state, for example.

In order to capture the underlying network topology, we must have that the state x_i, control u_i, and measurement y_i of node i can only depend on the states, control inputs, and measurements of the nodes in $N_p(i)$. To this end, we impose the following *sparsity* constraint on the matrices $\{A_p, B_p, C_p : p \in \mathcal{P}\}$:

$$j \notin N_p(i) \implies [A_p]_{ij} = [B_p]_{ij} = [C_p]_{ij} = 0, \quad p \in \mathcal{P}. \tag{2}$$

Under this constraint, the matrices A_p, B_p, C_p can be seen as an *encoding* of the topology of the graph $\mathcal{G}_p$. To demonstrate the sparsity constraint, consider the 3-node path graph shown in Fig. 1. For this graph, the matrix A_p must have the shown structure, where "$*$" can be any nonzero real number.

Further, in order to be able to design decentralized controllers, we must restrict the knowledge of node i about the graph $\mathcal{G}_p$. In particular, we assume that the knowledge of node i about the topology of $\mathcal{G}_p$ is only local; this can be captured by restricting the knowledge of node i to the set $\{[A_p]_i, [B_p]_i, [C_p]_i\}$. Formally, we make the following assumption.

Assumption 1: *The set $\mathcal{P}$ is finite, and the set $\{[A_p]_i, [B_p]_i, [C_p]_i : p \in \mathcal{P}\}$ is known to node i.*

Our goal is to design decentralized control inputs $[u]_i$, via an extension of the classical supervisory control scheme, in order to track the following stable linear reference model:

$$\begin{aligned} \dot{x}_m &= A_m x_m, \quad x_m(0) = x_m^0, \\ y_m &= C_m x_m, \end{aligned} \tag{3}$$

where $x_m \in \mathbb{R}^n$ and $A_m, C_m \in \mathbb{R}^{n\times n}$. Define the tracking error, e_T, as follows:

$$e_T = y_m - y.$$

The problem we are solving here is not the general tracking problem, because there is no external reference signal. The reason behind introducing the reference model is motivated by applications where the agents attempt to converge to a certain set-point *without the explicit knowledge of that point*. An example of such a scenario is the distributed averaging problem where nodes attempt to compute the average of their initial values, x_0, without knowing the value of the average a priori. We will apply our framework to the distributed averaging problem in Section V. Moreover, the standard stabilization problem, i.e., regulating the state x to the origin, is a special case of the problem we are solving and can be achieved by removing the reference model, i.e., setting $x_m \equiv 0$.

In the following section, we will introduce the distributed supervisory control scheme, and explain the functions of its main components in detail.

III. Distributed Supervisory Control Architecture

Fig. 2 illustrates the general architecture of the distributed supervisory control scheme. In this scheme, each node has access to a bank of candidate controllers that take as input the outputs of the nodes in its neighborhood as well as the tracking error. The "Sparse Filters" block in the figure emphasizes that the local dynamics and controllers of node i can only use information from neighboring nodes. It should be noted that there is no *centralized* sparse filter implemented, and this block is introduced for the sake of demonstration only. In this section, we will precisely explain how the information from the neighboring nodes affect the dynamics and control inputs of node i. Each node has a *local supervisor*: a dynamical system that takes as input the outputs and control inputs of the neighboring nodes and produces a *switching signal*. The switching signal provided by the supervisor activates one of the available controllers. The choice of a given control input is intended to minimize the tracking error. We will study the supervisor in more details next.

A. The Distributed Supervisor

We will refer to the collection of the local supervisors by the *distributed supervisor*. As illustrated in Fig. 3, the distributed supervisor has three main blocks: a multi-estimator, a monitoring signal generator, and a switching logic component. As in the centralized supervisory control case, there are certain properties we require from the individual blocks of the local supervisors which are crucial for achieving tracking. In particular, the multi-estimators must guarantee that at least one estimation error e_p is small. This will guarantee that

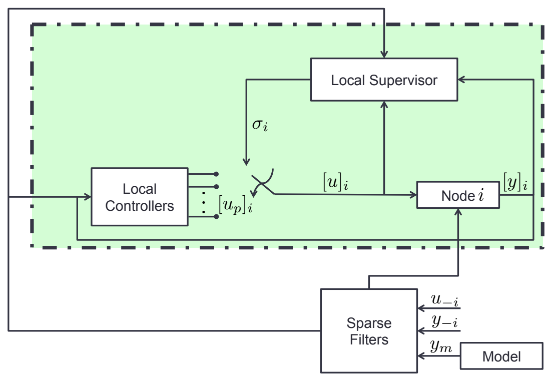


Fig. 2: Distributed supervisory control architecture.

switching halts in finite time. As for the candidate controllers, they must ensure that the closed loop system is detectable with respect to the estimation error. The switching logic must ensure that the estimation error is bounded, while avoiding fast switching. Here, we will work with a specific choice of these three blocks.

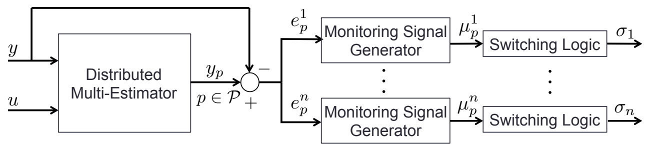


Fig. 3: The distributed supervisor.

Distributed Multi-Estimator and Candidate Controllers

For now, we assume that the control input u is given. We will explain how to select the control below. The distributed multi-estimator is a collection of local multi-estimators that are implemented at the nodes. At node i, the local multi-estimator is a dynamical system that takes as input the outputs and control inputs of the neighboring nodes, and it produces an estimate $[y_p]_i$, $p \in \mathcal{P}$. At each node, we adopt the standard Luenberger observer to design the multi-estimator. Let the matrix L_p be sparse:

$$j \notin N_p(i) \implies [L_p]_{ij} = 0. \tag{4}$$

The estimator equations at node i can then be written as

$$\begin{aligned} [\dot{x}_p]_i &= \sum_{j\in N(i)} [A_p]_{ij}[x_p]_j + [B_p]_{ij}[u]_j + [L_p]_{ij}[y_p - y]_j, \\ [y_p]_i &= \sum_{j\in N(i)} [C_p]_{ij}[x_p]_j, \end{aligned}$$

with arbitrary initial values $[x_p(0)]_i$. To write the estimator equations more compactly, let $x_p = [[x_p]_1, \ldots, [x_p]_n]^T$ and $y_p = [[y_p]_1, \ldots, [y_p]_n]^T$, for all $p \in \mathcal{P}$. Recalling that the matrices A_p, B_p, L_p, C_p are sparse, we can now write

$$\begin{aligned} \dot{x}_p &= A_p x_p + B_p u + L_p(y_p - y), \quad x_p(0) = x_p^0, \\ y_p &= C_p x_p, \end{aligned}$$

where $x_p^0 = [[x_p(0)]_1, \ldots, [x_p(0)]_n]^T$. It is important to note that x_p, y_p are not stored at any node in the network, since they are centralized quantities, and are introduced merely for notational simplicity.

We define the estimation error as $e_p = y_p - y$, $p \in \mathcal{P}$. We denote the estimation error at the i-th node by

$$e_p^i = [y_p - y]_i, \quad p \in \mathcal{P}.$$

As for the candidate control inputs at node i, we assume they are linear and given by

$$[u_p]_i = \sum_{j\in N(i)} [K_p]_{ij}[x_p]_j + [F_p]_{ij}[e_T]_j, \quad p \in \mathcal{P},$$

where the gain matrices K_p and F_p must be sparse to guarantee that the controllers are decentralized. Formally, we have the following constraint on the gain matrices:

$$j \notin N_p^i \implies [K_p]_{ij} = [F_p]_{ij} = 0, \quad p \in \mathcal{P}. \tag{5}$$

Similar to the estimators, for each $p \in \mathcal{P}$, we collect the control inputs of the nodes into the vector $u_p = [[u_p]_1, \ldots, [u_p]_n]^T$. We can then write

$$u_p = K_p x_p + F_p e_T, \quad p \in \mathcal{P}.$$

In general, the number of candidate control inputs need not be equal to $|\mathcal{P}| = r$. However, we will assume in this paper, for simplicity, that each node has access to r controllers.

Monitoring Signal Generators

Each node implements a monitoring signal generator which keeps track of the history of the estimation errors. This allows the switching decisions (to be explained next) to be based on the history of errors instead of the instantaneous estimation error values. The monitoring signals can be defined as any norm of the estimation error. Here, we define the monitoring signal at the i-th node as the square of the $\mathcal{L}_2$ norm of e_p^i. Formally, we write

$$\mu_p^i(t) = \int_0^t |e_p^i(s)|^2 ds. \tag{6}$$

It is more convenient for implementation purposes to express the monitoring signal as an ordinary differential equation (ODE):

$$\dot{\mu}_p^i = |e_p^i|^2, \quad \mu_p^i(0) = 0, \quad p \in \mathcal{P}.$$

Switching Logic

The switching logic at each node takes the monitoring signals μ_p^i, $p \in \mathcal{P}$, as inputs and produces a switching signal $\sigma_i : [0, +\infty) \to \mathcal{P}$ which determines the controller to be applied at each time instant. In particular, we have $[u]_i = [u_{\sigma_i}]_i$, $i \in \{1, \ldots, n\}$. The chosen controller should correspond to the monitoring signal that has the lowest value. However, if we set $\sigma_i = \min_{p\in\mathcal{P}} \mu_p^i$, we run into the risk of fast switching, which could be detrimental for the stability of the system [2]. To this end, we will employ hysteresis switching logic at each node with hysteresis constant $h_i > 0$. The hysteresis constant is introduced in order to prevent σ_i from switching its value too quickly. At each node, we first initialize the switching signal as follows:

$$\sigma_i(0) = \min_{p\in\mathcal{P}} \mu_p^i(0).$$

Let $\hat{p}_i(t) := \arg\min_{p\in\mathcal{P}} \mu_p^i(t)$. The signal σ_i switches its value at time t if $\mu^i_{\hat{p}_i} + h_i \leq \mu^i_{\sigma_i}$. Fig. 4 illustrates the hysteresis based logic at node i.

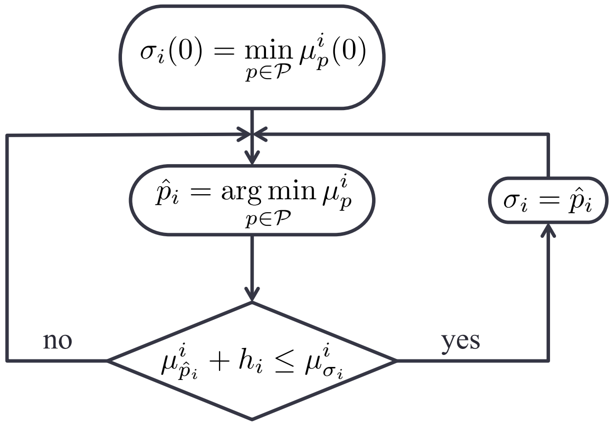

$\sigma_i(0) = \min_{p\in\mathcal{P}} \mu_p^i(0)$
$\hat{p}_i = \arg\min_{p\in\mathcal{P}} \mu_p^i$
$\sigma_i = \hat{p}_i$
no
$\mu^i_{\hat{p}_i} + h_i \leq \mu^i_{\sigma_i}$
yes

Fig. 4: Hysteresis based switching logic.

IV. Stability Analysis

In this section, we will obtain sufficient conditions for driving the tracking error to zero. Our approach will consist of two main steps. First, we will show that switching at all the nodes will halt in finite time. Then, assuming that the switching has stopped at all the nodes, we will study the detectability properties of the closed-loop system.

In order to prove that switching terminates in finite time, it is instrumental to show that $e_{p^\star}$ converges to zero exponentially fast. When $p = p^\star$, we have

$$\dot{x}_{p^\star} - \dot{x} = (A_{p^\star} + L_{p^\star} C_{p^\star})(x_{p^\star} - x).$$

To guarantee that $x_{p^\star}$ converges exponentially fast to x, we need to impose the following condition.

Condition 1: *The matrix $A_{p^\star} + L_{p^\star}C_{p^\star}$ is Hurwitz with $L_{p^\star}, C_{p^\star}$ satisfying (2) and (4), respectively.*

Remark 1: This condition can be viewed as a *distributed* version of detectability for the plant. In the case when $C_p = I$, for all $p \in \mathcal{P}$, this condition can be satisfied via diagonal dominance. Diagonal dominance can be achieved by choosing

$$[L_p]_{ii} < -[A_p]_{ii} - \max_{p\in\mathcal{P}} \sum_{j\neq i} |[A_p]_{ij}|.$$

Note that the maximization can be carried out locally at each node because of Assumption 1. To guarantee that L_p is sparse, we can select it to be a diagonal matrix. With such choice of L, the matrix $A_{p^\star} + L_{p^\star}$ becomes diagonally dominant with negative diagonal entries, and by Gershgorin's circle theorem, it follows that the matrix is Hurwitz. •

Under Condition 1, $x_{p^\star}$ converges exponentially fast to x, and consequently $e_p^\star = C_{p^\star}(x_{p^\star} - x)$ converges to zero exponentially fast regardless of the applied control u. We now have the following proposition, which is an immediate extension of its counterpart in the centralized architecture [2], [12].

Proposition 1: For all $i \in \{1, \ldots, n\}$, there exists a time $T_i^\star$ and an index $q_i^\star \in \mathcal{P}$ such that $\sigma_i(t) = q_i^\star$, for all $t \geq T_i^\star$. Moreover, $e^i_{q_i^\star} \in \mathcal{L}_2$, for all $i \in \{1, \ldots, n\}$.

Proof: Because $e_p^\star$ converges to zero exponentially fast, it follows from (6) that $\mu^i_{p^\star}$ is bounded. Let $K_i \in \mathbb{N}$ be such that $\mu^i_{p^\star} \leq K_i$. By definition, μ^i_p is a nondecreasing function, for all $p \in \mathcal{P}$. Hence, each μ^i_p must have a limit. Because $\mathcal{P}$ is finite, there exists a time T_i such that either $\mu^i_p \geq K_i$ or $\mu^i_p(t_2) - \mu^i_p(t_1) < h_i$ for all $t_2 > t_1 \geq T_i$; therefore, at most one more switch can occur for $t \geq T_i$. This in turn implies that there exists a time $T_i^\star$ such that $\sigma_i(t) = q_i^\star$, $q_i^\star \in \mathcal{P}$, for $t \geq T_i^\star$. Because $\mu^i_{p^\star}$ is bounded, $\mu^i_{q_i^\star}$ must also be bounded. By (6), it then follows that $e^i_{q_i^\star} \in \mathcal{L}_2$. ■

Note that after the switching stops, the estimate of node i, $q_i^\star$, might not match that of another node j, $q_j^\star$. In other words, the perception of node i about the underlying graph will in general be different than that of node j. This leads to new analysis challenges that were not present in the centralized structure.

In order to study the stability of the system following termination of switching, we first define

$$\begin{aligned} \hat{x}_{q^\star} &:= [[x_{q_1^\star}]_1, \ldots, [x_{q_n^\star}]_n]^T, \\ q^\star &:= [q_1^\star, \ldots, q_n^\star]^T. \end{aligned}$$

Further, we need to construct the following matrices:

$$\hat{A}_{q^\star} := \begin{bmatrix} [A_{q_1^\star}]_1 \\ \vdots \\ [A_{q_n^\star}]_n \end{bmatrix}, \hat{B}_{q^\star} := \begin{bmatrix} [B_{q_1^\star}]_1 \\ \vdots \\ [B_{q_n^\star}]_n \end{bmatrix}, \hat{C}_{q^\star} := \begin{bmatrix} [C_{q_1^\star}]_1 \\ \vdots \\ [C_{q_n^\star}]_n \end{bmatrix},$$

$$\hat{K}_{q^\star} := \begin{bmatrix} [K_{q_1^\star}]_1 \\ \vdots \\ [K_{q_n^\star}]_n \end{bmatrix}, \hat{F}_{q^\star} := \begin{bmatrix} [F_{q_1^\star}]_1 \\ \vdots \\ [F_{q_n^\star}]_n \end{bmatrix}, \hat{L}_{q^\star} := \begin{bmatrix} [L_{q_1^\star}]_1 \\ \vdots \\ [L_{q_n^\star}]_n \end{bmatrix}.$$

With these definitions, we can write the control law u after the switching stops as

$$u = \hat{K}_{q^\star}\hat{x}_{q^\star} + \hat{F}_{q^\star} e_T.$$

Define $\overline{x} := [x^T, \hat{x}_{q^\star}^T]^T$. After the switching stops, the closed-loop system becomes:

$$\begin{aligned} \dot{\overline{x}} &= \overline{A}\overline{x} + \overline{D}x_m \\ \hat{e}_{q^\star} &= \overline{C}\overline{x}, \end{aligned}$$

where

$$\overline{A} = \begin{bmatrix} A_{p^\star} - B_{p^\star}\hat{F}_{q^\star}C_{p^\star} & B_{p^\star}\hat{K}_{q^\star} \\ -(\hat{B}_{q^\star}\hat{F}_{q^\star} + \hat{L}_{q^\star})C_{p^\star} & \hat{A}_{q^\star} + \hat{B}_{q^\star}\hat{K}_{q^\star} + \hat{L}_{q^\star}\hat{C}_{q^\star} \end{bmatrix},$$

$$\overline{D} = \begin{bmatrix} B_{p^\star}\hat{F}_{q^\star}C_m \\ \hat{B}_{q^\star}\hat{F}_{q^\star}C_m \end{bmatrix},$$

$$\overline{C} = \begin{bmatrix} -C_{p^\star} & \hat{C}_{q^\star} \end{bmatrix}.$$

Consider now the matrix

$$\overline{\Gamma} = \begin{bmatrix} B_{p^\star}\hat{F}_{q^\star} + L_{p^\star} \\ \hat{B}_{q^\star}\hat{F}_{q^\star} + \hat{L}_{q^\star} \end{bmatrix},$$

and note that

$$\overline{A} - \overline{\Gamma}\,\overline{C} = \begin{bmatrix} A_{p^\star} + L_{p^\star}C_{p^\star} & B_{p^\star}(\hat{K}_{q^\star} - \hat{F}_{q^\star}\hat{C}_{q^\star}) - L_{p^\star}\hat{C}_{q^\star} \\ 0 & \hat{A}_{q^\star} + \hat{B}_{q^\star}(\hat{K}_{q^\star} - \hat{F}_{q^\star}\hat{C}_{q^\star}) \end{bmatrix}.$$

Using output injection, we can write

$$\dot{\overline{x}} = (\overline{A} - \overline{\Gamma}\,\overline{C})\overline{x} + \overline{\Gamma}\,\hat{e}_{q^\star} + \overline{D}x_m. \quad (7)$$

To achieve tracking, the matrix $\overline{A} - \overline{\Gamma}\,\overline{C}$ must be Hurwitz. Hence, in addition to Condition 1, we need to impose the following condition.

Condition 2: *The matrix $\hat{A}_{q^\star} + \hat{B}_{q^\star}(\hat{K}_{q^\star} - \hat{F}_{q^\star}\hat{C}_{q^\star})$ is Hurwitz for all $q^\star = [q_1^\star, \ldots, q_n^\star]^T$ with $\{q_1^\star, \ldots, q_n^\star\} \subset \mathcal{P}$, while satisfying (2) and (5).*

Remark 2: Assume that $B_p = C_p = I$, for all $p \in \mathcal{P}$, and let us select $[K_p]_i = -[A_p]_i$, for all i and p. Note that such selection for $[K_p]_i$ is made possible by Assumption 1. In this case, Condition 2 simplifies to requiring $-\hat{F}_{q^\star}$ to be sparse and Hurwitz. This can be achieved by selecting $F_p = kI$, where $k \in \mathbb{R}_{>0}$. •

We are now ready to state the main result of this section. Denote the state to which the reference model converges by $x_m^\star$.

Proposition 2: Under Conditions 1 and 2, and assuming that x_m converges asymptotically to $x_m^\star$, the state of the plant x remains bounded, and it asymptotically converges to

$$x^\star = -(A_{p^\star} + B_{p^\star}(\hat{K}_{q^\star} - \hat{F}_{q^\star}\hat{C}_{q^\star}))^{-1}B_{p^\star}\hat{F}_{q^\star}C_m x_m^\star.$$

Proof: Under Conditions 1 and 2, the matrix $\overline{A} - \overline{\Gamma}\,\overline{C}$ is Hurwitz. We know from Proposition 1 that $e^i_{q_i^\star} \in \mathcal{L}_2$ for all $i \in \{1, \ldots, n\}$. Noting that $\hat{e}_{q^\star} = [e^1_{q_1^\star}, \ldots, e^n_{q_n^\star}]$, we conclude that $\hat{e}_{q^\star}$ converges to zero as $t \to \infty$. Then, because x_m is bounded, we deduce from (7) that x must remain bounded. Using the fact that $\hat{e}_{q^\star}$ converges to zero, the steady-state expression follows immediately from (7). ■

Remark 3: Because the objective of the controller is to enable the plant to track the reference model, we are interested in cases where $x^\star = x_m^\star$. Assuming that B_p, C_p, C_m are all equal to the identity matrix, for all $p \in \mathcal{P}$, the steady-state expression simplifies to

$$x^\star = -(A_{p^\star} + \hat{K}_{q^\star} - \hat{F}_{q^\star})^{-1}\hat{F}_{q^\star}x_m^\star.$$

Hence, by setting $K_p = -A_p$ for all $p \in \mathcal{P}$, we will have $x^\star = x_m^\star$ if and only if $p^\star = q^\star$, i.e., when all the nodes correctly identify the unknown topology. Otherwise, there will be a discrepancy between x and the reference trajectory x_m. Nonetheless, in certain scenarios, this discrepancy may be negligible as we will demonstrate in Section V. •

Finally, we note that the multi-estimators and controllers we used here are only a specific possibility which we adopted to demonstrate the idea behind distributed supervisory control. One possible variation is to select the control inputs as

$$u_p = K_p y_p + F_p e_T, \quad p \in \mathcal{P}.$$

By following similar steps to the above, one can show that, with this choice of controllers, the matrix that is required to be Hurwitz in Condition 2 becomes

$$\hat{A}_{q^\star} + \hat{B}_{q^\star}(\hat{K}_{q^\star} - \hat{F}_{q^\star})\hat{C}_{q^\star}.$$

Hence, different choices of the controllers will provide different conditions on the system parameters to ensure stability. We are currently investigating different design choices that would place less restrictions on the system parameters.

V. Application: Tracking Consensus Dynamics

In this section, we apply the distributed supervisory control scheme to the distributed averaging problem [13], [14] in the case where the dynamics of the nodes contain unknown parameters. In distributed averaging networks, the nodes attempt to converge to the average of their initial values, $x(0)$, by performing local averaging. When the dynamics of the nodes contain unknown parameters, adaptive control techniques have been applied to solve this problem in [15]. By performing logic-based switching, our scheme enables convergence to the average without requiring continuous tuning of parameters as in the adaptive control approach. In [13], [16], the problem of achieving consensus when the underlying topologies are switching has been studied. Note that the topology in our case is unknown, but fixed, and the switching is performed at each node to choose the controller that minimizes the tracking error.

To specialize the reference model (3) to the distributed averaging dynamics, we assume that A_m is the negative of the weighted Laplacian matrix of a connected undirected graph. In particular, we have

$$\begin{aligned} &A_m = A_m^T \quad , \qquad A_m\mathbf{1} = 0, \\ &[A_m]_{ij} \geq 0 \quad , \qquad [A_m]_{ij} = 0 \iff (i,j) \notin \mathcal{E}, \quad i \neq j, \end{aligned}$$

where the weights $[A_m]_{ij}$, $j \neq i$ are randomly generated. The connectivity of the graph corresponding to A_m is necessary for the convergence to the average [13]. We assume that there is full state observation across the network; we therefore set $C_m = I$ and $C_p = I$, for all $p \in \mathcal{P}$. We also set $B_p = I$, for all $p \in \mathcal{P}$. Because the agents attempt to compute the average of their initial values, we set $x_m^0 = x_0$ and $x_p^0 = x_0$, for all $p \in \mathcal{P}$.

We consider a network of $n = 5$ agents and set $x_0 = [1, \ldots, 5]^T$. The agents will therefore attempt to converge to $\frac{1}{5}\mathbf{1}^T x_0 = 3$. We let $|\mathcal{P}| = 10$, that is, there are 10 possible topologies, and we set $p^\star = 10$. The matrices $\{A_p\}_{p\in\mathcal{P}}$ are generated at random, without any connectivity requirements.

In order to satisfy Condition 1, we pick $L_p = -kI$, for all $p \in \mathcal{P}$, where $k \in \mathbb{R}$ is selected as explained in Remark 1. In view of Remark 2, we set $K_p = -A_p$ and $F_p = 5I$, for all $p \in \mathcal{P}$, in order to satisfy Condition 2.

We will run two experiments, where we generate different $\{A_p\}_{p\in\mathcal{P}}$, A_m matrices, each time while respecting the connectivity constraint on A_m. Fig. 5 demonstrates the trajectories of the state of the network x, the state of the reference model x_m, the switching signals σ_i, and the tracking error e_T for the first experiment. In this case, all the agents correctly converge to the correct topology $\mathcal{G}_{p^\star}$, and, hence, converge to the average value 3. The tracking error therefore converges to zero.

Fig. 6 illustrates the same signals for the second experiment. In this case, agent 3 does not select the correct topology, i.e., $q_3^\star \neq 10$. Nonetheless, it converges to 3.09, and the tracking error is very small. The remaining nodes all converge to 3. A potential future research direction is quantifying the tracking error in the event where $q_i^\star \neq p^\star$.

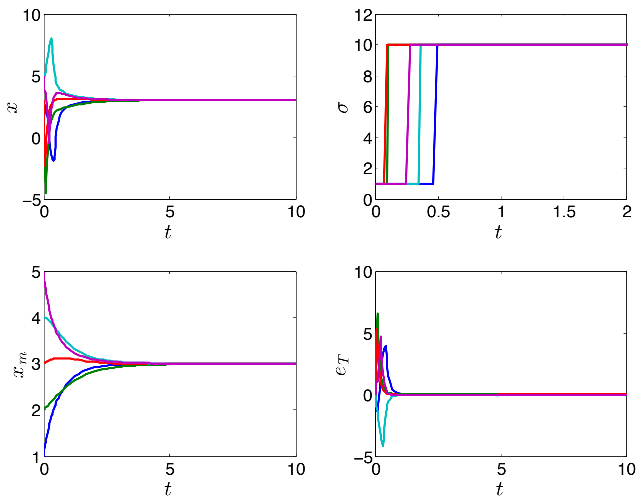


Fig. 5: All the agents correctly identify the unknown topology.

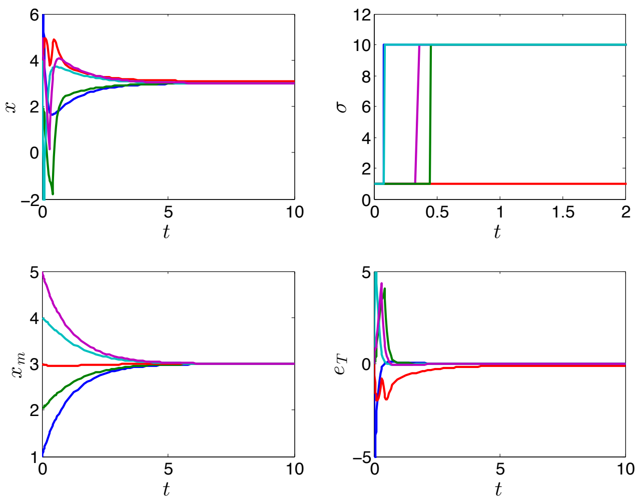


Fig. 6: One of the agents does not identify the correct topology.

VI. Conclusion

We proposed a distributed version of the classical centralized supervisory control scheme. Our scheme is based on logic-based switching among candidate controllers at each node. The switching decisions performed at each node depend only on information from neighboring nodes. The goal of the controllers is to track a set-point, without requiring the agents to have explicit knowledge of this point. The classical stabilization or regularization problem is a special case of this set-point tracking problem. We showed that switching stops in finite time at each node, and we provided sufficient conditions for stability. We applied our scheme to the distributed averaging problem when the dynamics of the agents contain unknown parameters. Simulation results demonstrated the efficacy of our scheme.

Future work will focus on making Condition 2 less strict, generalizing the problem to tracking of a reference model with a reference input signal, extending the scheme to handle time-varying unknown parameters, and designing incentive schemes to ensure that the majority of the agents identify the underlying network.

References

[1] J. P. Hespanha, D. Liberzon, and A. S. Morse, “Overcoming the limitations of adaptive control by means of logic-based switching,” *Systems & Control Letters*, vol. 49, no. 1, pp. 49–65, 2003.

[2] D. Liberzon, *Switching in systems and control*. Springer, 2003.

[3] A. S. Morse, “Supervisory control of families of linear set-point controllers part i. exact matching,” *Automatic Control, IEEE Transactions on*, vol. 41, no. 10, pp. 1413–1431, 1996.

[4] ——, “Supervisory control of families of linear set-point controllers. 2. robustness,” *Automatic Control, IEEE Transactions on*, vol. 42, no. 11, pp. 1500–1515, 1997.

[5] J. P. Hespanha, D. Liberzon, and A. S. Morse, “Logic-based switching control of a nonholonomic system with parametric modeling uncertainty,” *Systems & Control Letters*, vol. 38, no. 3, pp. 167–177, 1999.

[6] A. P. Aguiar and J. P. Hespanha, “Trajectory-tracking and path-following of underactuated autonomous vehicles with parametric modeling uncertainty,” *IEEE Transactions on Automatic Control*, vol. 52, no. 8, pp. 1362–1379, 2007.

[7] L. Vu, D. Chatterjee, and D. Liberzon, “Input-to-state stability of switched systems and switching adaptive control,” *Automatica*, vol. 43, no. 4, pp. 639–646, 2007.

[8] I. Al-Shyoukh and J. S. Shamma, “Switching supervisory control using calibrated forecasts,” *IEEE Transactions on Automatic Control*, vol. 54, no. 4, pp. 705–716, 2009.

[9] S. Baldi, G. Battistelli, E. Mosca, and P. Tesi, “Multi-model unfalsified adaptive switching supervisory control,” *Automatica*, vol. 46, no. 2, pp. 249–259, 2010.

[10] L. Vu and D. Liberzon, “Supervisory control of uncertain linear time-varying systems,” *IEEE Transactions on Automatic Control*, vol. 56, no. 1, pp. 27–42, 2011.

[11] ——, “Supervisory control of uncertain systems with quantized information,” *International Journal of Adaptive Control and Signal Processing*, vol. 26, no. 8, pp. 739–756, 2012.

[12] J. P. Hespanha, “Logic-based switching algorithms in control,” Ph.D. dissertation, Yale University, 1998.

[13] R. Olfati-Saber and R. M. Murray, “Consensus problems in networks of agents with switching topology and time-delays,” *IEEE Trans. Automat. Contr.*, vol. 49, no. 9, pp. 1520–1533, 2004.

[14] V. D. Blondel, J. M. Hendrickx, A. Olshevsky, and J. N. Tsitsiklis, “Convergence in multiagent coordination, consensus, and flocking,” in *Proc. Joint 44th IEEE Conf. Decision and Control and European Control Conf.*, December 2005.

[15] J. Yao, D. J. Hill, Z.-H. Guan, and H. O. Wang, “Synchronization of complex dynamical networks with switching topology via adaptive control,” in *Decision and Control, 2006 45th IEEE Conference on*. IEEE, 2006, pp. 2819–2824.

[16] W. Ren and R. W. Beard, “Consensus seeking in multiagent systems under dynamically changing interaction topologies,” *IEEE Transactions on Automatic Control*, vol. 50, no. 5, pp. 655–661, 2005.