Distributed Supervision Strategies for Cyber-Physical Systems With Varying Network Topology

Francesco Tedesco, *Senior Member, IEEE*, and Alessandro Casavola

***Abstract*—This article presents a novel distributed constrained monitoring strategy for cyber-physical systems that focuses on enforcing point-in-time set-membership coordination constraints among the subsystem evolutions. The proposed scheme extends previous solutions by allowing the handling of time-varying dynamic interconnections, including the online addition/removal of subsystems (plug-and-play) during normal system operations. This approach ensures global coordination of the subsystems while effectively handling corresponding changes in the underlying constraint topology. The coordination is achieved by determining a distributed feasible set-point for each subsystem when the nominal set-point becomes unfeasible due to topological changes in the system, resulting in constraints violation. In order to address this challenge, this article generalizes the distributed command governor (CG) theory to handle online requests for structural changes in the system and/or in the coordination constraints. An add-on module is provided for each local CG to process the coordination constraints and appropriately schedule the topology changes when certain formal conditions are satisfied. This ensures that the coordination constraints are satisfied even during the topological changes. Case studies are presented by evaluating the proposed strategy on the coordination of a formation of moving vehicles and a water distribution network. These simulations are used to evaluate the effectiveness and performance of the approach.**

***Index Terms*—Cyber physical systems, command governor, distributed reference management, plug and play control, varying network topology.**

Nomenclature

N	Number of interconnected subsystems in the CPS.
$\mathcal{A}$	Set of subsystems $\{1,\dots,N\}$.
Σ_i	ith subsystem model.
$x_i(t)$	State vector of ith subsystem Σ_i.
Φ_{ii}	Dynamic matrix in Σ_i.
G_i	Input map in Σ_i.
$g_i(t)$	CG reference for ith Σ_i.
Φ_{ij}	Coupling matrix Σ_i and Σ_j.
$y_i(t)$	Output vector of Σ_i.
$c_i(t)$	Constrained output of Σ_i.
Σ	Aggregate system collecting all Σ_is.
$x(t)$	State vector of Σ.
$g(t)$	Reference vector of Σ.
$y(t)$	Output vector of Σ.
$c(t)$	Constrained vector of Σ.
Φ	Dynamic matrix of Σ.
G	Input map of Σ.
H^y	Output map of Σ.
H^c	Constrained output map of Σ.
L	Input–output map of Σ.
$\mathcal{C}$	Compact and convex set describing the constraints for Σ.
$\mathcal{Z}$	Output admissible set related to the set $\mathcal{C}$.
$f_i(\cdot)$	Inequality function defining the output admissible set $\mathcal{Z}$.
$\mathcal{T}$	Turn, a subset of nonneighboring nodes.
$x_{[i]}$	Vector collecting the local states of the ith subsystem and its neighbors.
$g_{[i]}$	Vector collecting the local reference vectors of the ith subsystem and its neighbors.
$\tilde{x}_i$	Vector collecting the states of the neighbors of the ith agent.
$\tilde{g}_i$	Vector collecting the reference vectors of the neighbors of the ith agent.
$f_{[i]}(\cdot)$	Constraint function involving agent i and its neighbors.
$\mathcal{Z}_{[i]}$	Output admissible set related to the local constraints of agent i and its neighbors.
Φ^{new}	New matrix representing the modified system dynamics.
$\mathcal{Z}_{[i]}^{\text{new}}$	New output admissible set related to the modified local constraints of agent i and its neighbors.
$\mathcal{N}_i^{\text{new}}$	Set of new neighbors of agent i after the modification.

Manuscript received 22 July 2023; accepted 28 January 2024. Date of publication 6 February 2024; date of current version 30 July 2024. This work was supported in part by the research project under Grant 20225MESZB "Resilient Optimization in Distributed Cyber-Physical Control Problems subject to Adversarial Behaviour," in part by the Italian Ministry of University and Research (MUR), within the PRIN 2022 program, and in part by the European Union—Next Generation EU. Recommended by Senior Editor T. Iwasaki. *(Corresponding author: Francesco Tedesco.)*

The authors are with the Dipartimento di Elettronica, Informatica e Sistemistica, Università della Calabria, 87036 Rende (CS), Italy. (e-mail: francesco.tedesco@unical.it; a.casavola@dimes.unical.it).

Digital Object Identifier 10.1109/TAC.2024.3362884

I. Introduction

In LAST years, the control and coordination of multiagent networked systems has emerged as a topic of increasing interest in the control community [1], [2], [3], [4], [5]. This surge in interest can be attributed in part to the diverse array of

applications currently under investigation, such as unmanned air/ground/underwater vehicles, automated highway systems, power grids, telecommunication systems, water distribution systems, and more.

A recent appellation used by experts for these multiagent paradigms is cyber-physical systems (CPSs) [6], [7]. The name CPSs reflects the dynamic interactions between computers, networking media/resources, and physical systems, necessitating multidisciplinary methodologies and technologies (embedded systems, computers, communications, and controls) to collaborate and accomplish the prescribed mission [8].

In view of this, the development of distributed control and supervision techniques is typically preferred over centralized approaches, as the latter often become impractical for large-scale spatially distributed systems due to unrealistic computational and communication requirements. While modern control and supervision schemes offer advantages, such as scalability and modularity, their design is generally more challenging than their centralized counterparts. In fact, they require specific protocols to coordinate the actions of multiple local computing units, referred to as *agents*, that interact within a shared environment to achieve a local/common goal. The complexity increases when dealing with dynamically interconnected subsystems, as the control problems to be locally solved by agents often exhibit appreciable coupling among them. Consequently, agents involved in a coupled control or supervision problem are typically designated as *neighbors* and require special attention.

In the case of large-scale CPS, managing potential variations in the dynamical coupling structure of the system becomes essential when subsystems are added or removed. This scenario frequently arises in vast infrastructures of interacting subsystems, each composed of complex systems, such as transportation or distribution networks, and sometimes referred to as system-of-systems [9]. In such cases, the supervision of these agents must possess the capability to reconfigure their behavior in response to changing requirements.

A. Literature Review

Traditional distributed control approaches (e.g., [10], [11]) often lack the necessary flexibility to adapt to system changes and typically necessitate a complete redesign of the control system. Specifically, as demonstrated in the second example in Section VI, sudden modifications in the plant structure without a proper reconfiguration in the control logic can lead to constraints violation and instability in the worst case.

On the other hand, replanning the control system from scratch can be impractical due to the associated costs related to system shutdown and startup processes. In order to tackle this challenge, several contributions in the literature have focused on handling time-varying couplings within distributed control and supervision frameworks. Notably, distributed model predictive control (MPC) approaches [12], [13], [14], [15], [16], [17] have been of interest for our purposes.

Particularly, some computational demanding hierarchical schemes [13] (limited to dynamically decoupled systems), [14], and iterative methods [12] have been introduced to enhance the control performance by addressing the problem of the online modification of either the priority of command update among the agents or the communication links. However, these methods do not allow for the modification of constraints, especially online.

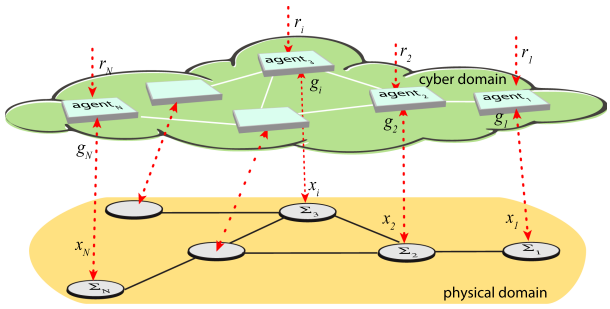

Fig. 1. CPS multiagent architecture based on CG strategy: Dynamic couplings (black lines), constraints couplings (white lines).

In contrast, none of the cited papers address the concept of plug-and-play (PnP) control, which aims at automatically reconfiguring the controller when system components are plugged in or removed. The problem of PnP control has been addressed in [15], [16], [17], where noniterative distributed MPC approaches have been carefully designed to accommodate changes in the system arising from the addition or removal of one or more subsystems during closed-loop operations.

B. Contribution

In this study, we investigate the technical issues, mentioned previously, within the class of distributed CPSs depicted in Fig. 1. The physical layer of these systems comprises dynamically coupled systems equipped with decentralized controllers that ensure the asymptotic stability of the entire network. In that Figure, the generic ith agent in the *cyber* domain is responsible for modifying the nominal reference signal $r_i(t)$ to its best feasible approximation $g_i(t)$ whenever direct application of r_i to the ith subsystem would result in constraints violation. This modification is based on $x_i(t)$ and other available information about the neighborhoods. Subsequently, the local supervised subsystem Σ_i is fed by the feasible reference $g_i(t)$, and this process is repeated at each time instant for all agents.

This approach is inspired by the well-known command governor (CG) approach [18], [19]. However, our study aims to generalize its action to manage requests for variations in the subsystems' topology and/or constraints' structure during the online operations. To address these scenarios, we have designed agents in the *cyber* domain by using a novel bilevel supervision architecture, where the lower level implements a noniterative, noncooperative distributed CG algorithm, grouping agents with different constraints into particular sets (turns) [11]. These agents update their control actions simultaneously based on local information only. On the other hand, the higher level is responsible for two main tasks: the distributed reconfiguration of the local CGs and the redetermination of the turns configuration in response to requests for *topological* variation.

This new proposed architecture presents differences, original points and new insight with respect to [18] and the existing literature on distributed MPC. In particular, the following three main contributions are worth being mentioned.

1) A novel distributed supervision scheme able to deal with time-varying constraints and PnP operations at the same time and in a rigorous way.
2) Guarantees on the accomplishment of finite-time topological modifications during the normal online operations while preserving the stability of the entire network and the constraints fulfillment.
3) A modular stack layered architecture for modeling the CPS under investigation.

Point 1) requires a generalization of the standard CG theory in order to address the distributed reference management problem in the presence of time-varying subsystem interconnections and constraint variations that may occur during the normal operations. Existing contributions in this area have primarily focused on the centralized case and specific applications (e.g., [23]). However, the distributed case poses more significant challenges to be faced. In contrast to the previously cited works [12], [13], [14], [15], [16], [17], which focus on distributed MPC schemes, our distributed CG-based approach is distinctive as it does not require conveying future state trajectories or control sequences. Instead, it only needs the exchange of current local state $x_i(t)$ and applied command $g_i(t)$ among neighbors. Moreover, none of the above-mentioned papers simultaneously addresses both global constraints modifications and PnP operations. PnP problems differ from other important cases, such as changes in the local structure or interconnections of subsystems, or modifications of existing dynamical couplings. In this article, we demonstrate that PnP operations can be effectively addressed as a specific case of constraints modification.

Point 2) addresses a significant technical challenge concerning PnP operations, which involves ensuring that a PnP request can always be satisfied within a finite time. Generally, successful completion of a PnP operation relies on certain conditions being met, which are necessary to guarantee stability and constraints fulfillment based on the current states of the involved subsystems. Existing approaches [15], [16], [17] envisage a denial of PnP requests if these conditions do not hold true. In such cases, the PnP request is renewed after a random number of steps, without any assurance that the operation will eventually succeed in the future. In contrast, this article proposes an iterative distributed procedure running among agents to possibly bypass such denials. This procedure temporarily modifies the nominal reference signals for a minimal number of agents, creating a safe environment for a topological change. To achieve this, we determine formal conditions crucial for preserving the feasibility of the entire network during configuration switchings. In addition, we devise procedures inspired by graph-colorability concepts [31] to redetermine the turns after a topological change. These investigations constitute key aspects of our study.

Finally, while the primary contribution of this article does not directly focus on the foundational theory or modeling of CPSs, as explored in [20], [21], Point 3) can be regarded as a novel attempt to address a distributed reference management problem within the CPS domain. In recent years, there has been a proliferation of modeling approaches and tools dedicated to handling the inherent heterogeneity in CPSs and capturing domain-specific concepts and requirements [22]. In this context, the stack layered structure presented in this article offers novelty and seems well-suited for representing a distributed CPS when dealing with reference management issues. The design flexibility of the stack architecture, especially its topology layer, allows for modular inclusion of features without necessitating a complete redesign of existing control architectures implementing the distributed CGs.

C. Outline

The rest of this article is organized as follows. A preliminary section reports some notations, definitions on distributed optimization and graph theory. In Section III, the problem of interest is stated after presenting the physical modeling of the CPS under investigation. Section IV is devoted to describe the proposed supervision architecture of the distributed CPS. In Section V, the main properties of the proposed scheme have been presented. They include the recursive feasibility and stability properties that are formally proved. Hence, two illustrative examples focused respectively on dynamically decoupled and coupled systems are presented in Section VI for assessment purposes. Finally, Section VI concludes this article.

Preliminary versions of this work have been presented in [24], where the above-mentioned problem has been considered for the simpler case of decoupled systems, and in [25] where dynamical coupling amongst the physical subsystems is allowed. Anyway, this article represents a significant added value because of the following aspects.

1) The problem to be solved is introduced in a more formal and comprehensive way by introducing several assumptions (**A2, A3, A4**) and notions before the problem statement.
2) Relevant details are revealed about each stage of the working logic of the proposed supervision architecture in algorithmic form (Algorithms 2 and 3) and by means of specific state automata.
3) The stability of the whole system under dynamic topology switching occurrences have been analyzed in Theorem 2 within Section V.
4) All the theoretical results contained in lemmas, propositions, and theorems are here endowed with formal proofs and have been presented in a more comprehensive fashion.
5) The simulation section has been enriched with an example involving a significant large scale system.

II. Notations and Preliminaries

$\mathbb{R}$, $\mathbb{R}_+$, and $\mathbb{Z}_+$ denote the real, nonnegative real, and nonnegative integer numbers, respectively. The Euclidean norm of a vector $x \in \mathbb{R}^n$ is denoted by $\|x\| = \sqrt{x_1^2 + \cdots + x_n^2}$ whereas $\|x\|_\Psi^2$, $\Psi = \Psi^T > 0$, denotes the quadratic form $x^T \Psi x$. For given sets $\mathcal{A}, \mathcal{E} \subset \mathbb{R}^n$, $\mathcal{A} \sim \mathcal{E} := \{a : a + e \in \mathcal{A}, \forall e \in \mathcal{E}\}$ is the *Pontryagin set difference*, while $\mathcal{A} \oplus \mathcal{E} := \{z = a + e :$

$\forall a \in \mathcal{A}, \forall e \in \mathcal{E}\}$ denotes the *Minkowski sum*. Given a set $S \subseteq X \times Y \subseteq \mathbb{R}^n \times \mathbb{R}^m$, the projection of S onto X is defined as $\text{Proj}_X(S) := \{x \in X \mid \exists y \in Y \text{ s.t. } (x, y) \in S\}$.

Given a matrix $A \in \mathbb{R}^{n\times n}$, let $\lambda_1, \ldots, \lambda_n$ be the eigenvalues of A. Then, A is Schur if $\max_i |\lambda_i| < 1$.

Definition 1. [Pareto optimal (PO) solution]: Consider the following multiobjective problem:

$$\min_g [f_1(g_1), f_2(g_2) \ldots, f_N(g_N)]$$

$$\text{subject to } g = [g_1^T, \ldots, g_i^T, \ldots, g_N^T]^T \in \mathcal{S}.$$

The vector $g^{*_p} \in \mathcal{S}$ is a PO solution if there exist no other $g \in \mathcal{S}$ such that: $f_i(g_i) \leq f_i(g_i^{*_p}), i = 1, \ldots, N$ for which there exists a j such that $f_j(g_j) < f_j(g_j^{*_p})$.

Definition 2. (Admissible direction): Let $\mathcal{S} \subset \mathbb{R}^m$ be a convex set and consider an arbitrary point $g \in \mathcal{S}$. The vector $v \in \mathbb{R}^m$ represents an admissible direction for $g \in \mathcal{S}$ if there exists a positive scalar $\bar{\lambda} > 0$ such that $(g + \lambda v) \in \mathcal{S}, \lambda \in [0, \bar{\lambda}]$.

Definition 3. (Decision set of agent i) [28]: The decision set $\mathcal{V}_i^{\mathcal{S}}(g)$ of the ith agent at a point $g \in \mathcal{S}$ is the set of all admissible directions along which it can move, under the assumption that all other agents are maintaining unchanged their commands, in updating its action viz. $\mathcal{V}_i^{\mathcal{S}}(g) := \{v \in \mathrm{IR}^{m_i} : [0_1^T, \ldots, 0_{i-1}^T, v^T, 0_{i+1}^T, \ldots, 0_N^T]^T$ is an admissible directionfor $g \in \mathcal{S}\}$.

Definition 4. (Viability property) [28]: A point $g \in \mathcal{S}$ is "*viable*" if, for any admissible direction $v = [v_1^T, \ldots, v_N^T]^T \in \mathbb{R}^{\mathrm{m}}$, $v_i \in \mathbb{R}^{\mathrm{m_i}}$ with $\sum_{i=1}^N m_i = m$, there exists at least one agent i such that the subvector $v_i \neq 0$ belongs to its decision set, i.e., $v_i \in \mathcal{V}_i^{\mathcal{S}}(g)$.

Definition 5. (Graph): A graph is an ordered pair $\Gamma(\mathcal{A}, \mathcal{E})$, such that:

- $\mathcal{A}$ is the set of nodes;
- $\mathcal{E}$ is a subset of pairs of $\mathcal{A}$ known as the set of edges connecting two nodes, i.e., $\mathcal{E} \subset \mathcal{A} \times \mathcal{A}$.

Definition 6. (Degree of a node): Given a graph $\Gamma(\mathcal{A}, \mathcal{E})$, $\Delta_i(\Gamma) : \mathcal{A} \to \mathbb{Z}_+$ is the number of edges incident to a node.

Definition 7. (Degree of a graph): Given a graph $\Gamma(\mathcal{A}, \mathcal{E})$, $\Delta(\Gamma) := \max_{v\in\mathcal{A}} \Delta_v(\Gamma)$ is the degree of the graph.

Definition 8. (Neighborhood of the ith node): The neighborhood $\mathcal{N}_i$ of the ith node in $\Gamma(\mathcal{A}, \mathcal{E})$ consists of all its adjacent nodes i.e.,

$$\mathcal{N}_i = \{i\} \cup \{j \in \mathcal{A} : (i, j) \in \mathcal{E}\}. \tag{1}$$

III. System Description and Problem Formulation

The physical component of the CPS under investigation is composed by a set of N interconnected subsystems $\mathcal{A} = \{1, \ldots, N\}$. Each ith subsystem is modeled by means of the following difference equations:

$$\Sigma_i : \begin{cases} x_i(t+1) = \Phi_{ii} x_i(t) + G_i g_i(t) + \sum_{j\in\mathcal{A}\setminus\{i\}} \Phi_{ij} x_j(t) \\ y_i(t) = H_i^y x_i(t) \\ c_i(t) = H_i^c x_i(t) + L_i g_i(t) \end{cases} \tag{2}$$

where $t \in \mathbb{Z}_+$, $x_i \in \mathbb{R}^{n_i}$ is the state vector (which includes the controller state under dynamic regulation), $r_i \in \mathbb{R}^{m_i}$ is the nominal reference vector, $g_i \in \mathbb{R}^{m_i}$ a feasible reference vector provided by the CG that, if no constraints (and no CG) were present, would coincide with the desired reference $r_i \in \mathbb{R}^{m_i}$, and $y_i \in \mathbb{R}^{m_i}$ is the output vector related to the tracking performance ($y_i(t) \approx r_i(t)$). Furthermore, $c_i \in \mathbb{R}^{n_c^i}$ is the constrained vector that collects all constrained local variables. From a global point of view, the entire physical part of the CPS to be designed can be described by the following aggregate system Σ having the form:

$$\Sigma : \begin{cases} x(t+1) = \Phi x(t) + G g(t) \\ y(t) = H^y x(t) \\ c(t) = H^c x(t) + L g(t) \end{cases} \tag{3}$$

with $x(t) = [x_1^T(t), \ldots, x_N^T(t)]^T, g(t) = [g_1^T(t), \ldots, g_N^T(t)]^T$, $y(t) = [y_1^T(t), \ldots, y_N^T(t)]^T, c(t) = [c_1^T(t) \ldots, c_N^T(t)]^T, \Phi = [\Phi_{ij}]_{i,j=1,\ldots,N}, G = \text{diag}(G_1, \ldots, G_N), H^y = \text{diag}(H_1^y, \ldots, H_N^y), H^c = \text{diag}(H_1^c, \ldots, H_N^c), L = \text{diag}(L_1, \ldots, L_N)$. It is assumed that each subsystem Σ_i represents a closed-loop dynamic regulated by a decentralized local controller in a way that the aggregated system Σ is asymptotically stable, in other words it is assumed that:

(A1) Φ *is Schur.*

In this context, the cyber layer of the CPS under analysis, to be designed in the next section, is devoted to deal with the following distributed CG design problem

Problem 1: Given systems (2), locally determine, at each time instant t and for each agent $i \in \mathcal{A}$, a suitable reference signal $g_i(t)$ that is the "best approximation" (to be better specified later) of $r_i(t)$ (according to a selection index specified in the following) and such that its application does not produce constraints violation, i.e.,

$$c(t) \in \mathcal{C}, \quad \forall t \in \mathbb{Z}_+ \tag{4}$$

with $\mathcal{C}$ being a compact and convex set described by a list of q_c inequalities, i.e.,

$$\mathcal{C} := \{c \in \mathbb{R}^{n_c} : l_h(c) \leq 0, \quad h = 1, \ldots, q_c\}$$

where $n_c := \sum_{i=1}^N n_c^i$. Note that $\mathcal{C}$, besides local constraints, can also describe global constraints arising among the variables of all subsystems.

A. Constraints Structure

In this section, some definitions are introduced to deal with the constraints arising in the context of distributed CG-based supervision strategies. More in detail, let

$$\begin{aligned} x_g &:= (I - \Phi)^{-1} G g \\ y_g &:= H^y (I - \Phi)^{-1} G g \\ c_g &:= H^c (I - \Phi)^{-1} G g \end{aligned} \tag{5}$$

represent the steady-state solutions of the system to a constant g (3) while

$$c(k,x,g) := H^c\left(\Phi^k x + \sum_{\tau=0}^{k-1}\Phi^{k-\tau-1}Gg\right) + Lg \tag{6}$$

are the virtual predictions of the constrained vector c over the *virtual* time k when a constant command sequence $g(k) \equiv g \forall k$, is applied to Σ starting from the initial aggregate state $x(0) = x$.

Then, the notion of *output admissible set* (OAS) related to the set $\mathcal{C}$ is of interest

$$\mathcal{Z} := \left\{(x,g)\in\mathbb{R}^{n\times m} \;\middle|\; \begin{array}{l} c_g \in \mathcal{C} \sim \mathcal{B}_\delta, \\ c(k,x,g) \in \mathcal{C},\ \forall k \in \mathbb{Z}_+ \end{array}\right\} \tag{7}$$

where $\mathcal{B}_\delta := \{x \in \mathbb{R}^{n_c} : \|x\| \le \delta\}$ is a generic ball centered in 0_x of $\mathbb{R}^{n_c}$. It can be proved, see e.g., ([32]), that i) the constraints (4) are always satisfied if

$$(x(t), g(t)) \in \mathcal{Z},\ \forall t \in \mathbb{Z}_+ \tag{8}$$

and ii) the above-mentioned set $\mathcal{Z}$ is finitely determined and can be formulated by n_q inequalities $f_i : \mathbb{R}^{n\times m} \to \mathbb{R}$, i.e.,

$$\mathcal{Z} := \left\{(x,g) \in \mathbb{R}^{n\times m} \,|\, f_h(x,g) \le 0,\ h = 1,\dots,n_q\right\} \tag{9}$$

with $n_q \ge n + \sum_{k=1}^{N} n_c^i$.

In the previous definition, it is worth noting that in some cases each function f_h in $\mathcal{Z}$ involves only a restricted subset of agents. This makes sense when the subsystems are characterized by weak dynamic interactions and few coupling constraints. In view of this, it is convenient to relate the structure of the constraints (9) to an indirectly connected graph $\Gamma(\mathcal{A},\mathcal{E})$, whose set of nodes coincides with the set of agents $\mathcal{A}$, and where $\mathcal{E} \subset \mathcal{A}\times\mathcal{A}$ is the set of edges connecting the agents (nodes) whose subsystem evolutions are jointly constrained in (9), i.e.,

$$\mathcal{E} := \{(i,j) : \text{both } i \text{ and } j \text{ appears in at least one } f_h \text{ in (9)}\}. \tag{10}$$

Then, the definition of *turn* can be introduced.

Definition 9. (Turn): A turn $\mathcal{T} \subset \mathcal{A}$ is a *subset of nonneighboring nodes*, i.e., $\forall i,j \in \mathcal{T}$ such that $i \ne j$, $j \notin \mathcal{N}_i$ (none of them is a neighbor of the others).

The above-mentioned definition is equivalent to the well-known notion of *independent set* in graph theory [29]. From the perspective of the ith agent, the following vector becomes relevant:

$$x_{[i]} := [x_{i_1}, x_{i_2}, \dots, x_{i_h}, \dots, x_{i_{N_i}}] \in \mathbb{R}^{n_{[i]}} \tag{11}$$

with $n_{[i]} := \sum_{i\in\mathcal{N}_i} n_i$. The vector $x_{[i]}$ collects the local states $x_h, \forall h \in \mathcal{N}_i$, of the ith subsystem and of all its neighbors. Please notice that vectors $g_{[i]} \in \mathbb{R}^{m_{[i]}}$ and $x_{g_{[i]}}$ can be analogously defined, while vectors $\tilde{x}_i \in \mathbb{R}^{n_{[i]}-n_i}$, $\tilde{g}_i \in \mathbb{R}^{m_{[i]}-m_i}$ and $\tilde{x}_{\tilde{g}_{[i]}}$ collect, respectively, all states, references, and equilibria of the neighbors of the ith agent. In a similar way, let us recast each function f_h in (9) involving agent i and its neighbors only as

$$f_h(x,g) := f_{[i]}^h(x_{[i]}, g_{[i]}) + [0_1,\dots,0_n]^T x + [0_1,\dots,0_m]^T g \tag{12}$$

with $f_{[i]}^h : \mathbb{R}^{n_{[i]}\times m_{[i]}} \to \mathbb{R}$ representing the actual constraints. Within this perspective, $f_h(x,g)$ can be seen as the augmented version of $f_{[i]}^h(x_{[i]}, g_{[i]})$ in the extended space $\mathbb{R}^{n+m}$. Then, if for each agent i, we collect the related $f_{[i]}$s in the following set:

$$\mathcal{Z}_{[i]} := \left\{(x_{[i]}, g_{[i]}) \in \mathbb{R}^{n_{[i]}\times m_{[i]}} : f_{[i]}^h(x_{[i]}, g_{[i]}) \le 0, h = 1,\dots,n_{q_i}\right\} \subseteq \mathbb{R}^{m_{[i]}\times n_{[i]}} \tag{13}$$

with $n_{q_i} \le n_q$, the set (9) can be recast as an intersection of $|\mathcal{A}|$ sets, that is

$$\mathcal{Z} := \bigcap_{i\in\mathcal{A}} \left(\mathcal{Z}_{[i]} \times \mathbb{R}^{(n-n_{[i]})\times(m-m_{[i]})}\right). \tag{14}$$

B. Time-Varying Constraints and Problem Formulation

In this section, the previous basic distributed CG setup is generalized in order to consider possible requests for variation of the structure of the constraints, which are issued at a certain time $t_s > 0$ by an unspecified entity, which can be internal or external, to the agents of the network. In particular, this request is presented in advance to a single agent i according to the following scenarios.

Scenario 1. (Modification of system dynamic): One agent $i \in \mathcal{A}$ requires to modify either its local dynamic, i.e., $\Phi_{ii} \leftarrow \Phi_{ii}^{\text{new}}$ or the existing dynamic coupling with another agent $j \in \mathcal{A}$, i.e., $\Phi_{ij} \leftarrow \Phi_{ij}^{\text{new}}$.

Scenario 2. (Modification of constraints structure): One agent $i \in \mathcal{A}$ requires to modify some inequalities describing $\mathcal{C}$ involving its local constrained vector $c_i(t)$, i.e., $\mathcal{C} \leftarrow \mathcal{C}^{\text{new}}$ with $\mathcal{C}^{\text{new}}$ being a convex and compact set as well.

Furthermore, in Scenario 1 it is worth remarking that the new aggregate dynamic represented by the matrix Φ^{new} is assumed to be Schur as the original matrix Φ, more formally:

(**A2**) Φ^{new} *is Schur.*

Moreover, before stating the problem to be solved in a formal way it is important to assume that:

(**A3**) *Both Scenarios 1 and 2 translate into the request of changing the local set $\mathcal{Z}_{[i]}$ and, at most, the sets $\mathcal{Z}_{[j]}$ for some $j \in \mathcal{N}_i \cup \mathcal{N}_i^{new}$, i.e., $\mathcal{Z}_{[i]} \leftarrow \mathcal{Z}_{[i]}^{new} \Rightarrow \mathcal{Z}_{[j]} \leftarrow \mathcal{Z}_{[j]}^{new}$ for some $j \in \mathcal{N}_i^{new}$.*

Such an assumption naturally implies that the global set $\mathcal{Z}$ and the graph topology Γ must be modified to satisfy the request given in Scenarios 1 or 2. Conversely, it suggests that all relevant information about removing or adding inequalities in $\mathcal{Z}$ is contained in $\mathcal{Z}_{[i]}^{\text{new}}$. Consequently, the ith agent directly involved in this modification will be referred to as the *changing agent* throughout this article.

Finally, as far as the communication facilities are concerned, the following assumption is in order.

(**A4**) *Each agent has a direct communication link with its neighbors in $\mathcal{N}_i$ with whom it shares its current state $x_i(t)$ and command $g_i(t)$. Moreover, before accomplishing the operations requested in either Scenarios 1 or 2, each agent first establishes the communications with the possibly new neighbors that will belong to the set $\mathcal{N}_i^{new}$ once those operations will be completed.*

Then, the above-mentioned considerations lead to the definition of this further Problem 2 to be solved within the *cyber* domain described in the next section.

Problem 2: Given the constrained system (2), determine a turn-based (TB) supervisory strategy able to deal with Problem 1 and, whenever either Scenarios 1 or 2 occur, capable of determining local references $g_i(t)$ that guarantee a safe transition from $\mathcal{Z}$ to $\mathcal{Z}^{\text{new}}$ in a finite time.

A viable solution to Problem 2 requires the design of a supervision scheme that adheres to the following principles.

1) Each individual agent uses only local data and data coming from its neighbors when updating its commands.
2) In the case of either Scenarios 1 or 2 taking place, a systematic procedure can be implemented that includes the following tasks. i) Assessing whether the operation affects the overall system constraint fulfillment. If so, the affected agents are automatically reconfigured through a process called the *constraint reconfiguration (CR)* task. ii) Accomplishing the online redetermination of the turns in response to possible changes in the constraint topology that may be induced on the graph Γ, as a result of the operation. This procedure is hereafter referred to as the *online turn determination (OTD)* task.

Remark 1: Please note that Scenario 1 can also involve PnP operations related to the addition or removal of a subsystem Σ_i. Specifically, a *plug-in* operation refers to the situation where a subsystem Σ_i, initially having no dynamic couplings with other subsystems and no conflicting constraints [it is essentially a new node i with an empty neighborhood in $\Gamma(\mathcal{A}, \mathcal{E})$], is required to join some other subsystems by incorporating new dynamic coupling terms of the form $\Phi_{ij}x_j(t)$ for certain $j \in \mathcal{A} \setminus i$. This necessitates the replacement of the current OAS $\mathcal{Z}$ with a new one containing additional inequalities, leading to the addition of new edges in Γ. Conversely, in a *plug-out* operation, a subsystem Σ_i requests the removal of coupling terms $\Phi_{ij}x_j(t)$ for all $j \in \mathcal{N}_i$ from its dynamics. In this case, certain inequalities in $\mathcal{Z}$ may disappear or require modification, resulting in the removal of all edges associated with i from Γ.

Remark 2: Note that the origin of a constraint modification request is strictly dependent on the type of application. For example, in a network of decoupled systems, such as autonomous vehicles, constraints modification (e.g., formation switching) may be activated by an internal self-request from a specific *changing agent*. On the other hand, external requests may come from agents belonging to other networks, or from human users in the case of dynamically coupled systems where constraint modification involves human intervention in the physical layer of the system. In all these cases, we assume that the *changing agent* receiving the modification request will be provided with the new $\mathcal{Z}^{\text{new}}_{[i]}$. After that, all procedures aimed at solving Problem 2 can start. However, specific details about the computation of $\mathcal{Z}^{\text{new}}_{[i]}$ have been omitted because it would require distinguishing among several possible coupling configurations that the new set introduces. Therefore, it is not possible to determine a priori which information is required, as it strongly depends on the number of agents involved in $\mathcal{Z}^{\text{new}}_{[i]}$.

C. Conditions for Switching Among Time-Varying Constraints

This section contains part of the main contribution of this work because it presents local conditions to guarantee a safe transition among two global OASs $\mathcal{Z}$ and $\mathcal{Z}'$. To this end, the notion of *switchable* state is in order.

Definition 10: Let $x \in \mathbb{R}^n$ be a state related to system Σ and $\mathcal{Z}, \mathcal{Z}'$ two different OASs. Then, the state x is said $(\mathcal{Z}, \mathcal{Z}') - switchable$ if there exists a command $g \in \mathbb{R}^m$ such that the pair $(x, g) \in \mathcal{Z} \cap \mathcal{Z}'$. In this case, the pair $(\mathcal{Z}, \mathcal{Z}')$ is said *replaceable*.

The latter definition is related to global quantities. However, both Scenarios 1 and 2 operations affect the constraints structure of at most one agent i and its neighbors. In view of this, it is convenient, for the successive developments, to point out that $\mathcal{Z}^{\text{new}}$ arises from an intersection between $\mathcal{Z}$ and a set involving $\mathcal{Z}^{\text{new}}_{[i]}$, i.e.,

$$\mathcal{Z}^{\text{new}} \supseteq \mathcal{Z} \cap \left(\mathcal{Z}^{\text{new}}_{[i]} \times \mathbb{R}^{(n-n^{\text{new}}_{[i]})\times\left(m-m^{\text{new}}_{[i]}\right)} \right). \tag{15}$$

Then, the following local conditions for the *switchability* of a state can be provided.

Theorem 1: Let $\mathcal{Z}$ and $\mathcal{Z}^{\text{new}}$ be two *replaceable* sets satisfying (15) for a certain agent i involved in either Scenarios 1 or 2 with related set $\mathcal{Z}^{\text{new}}_{[i]}$ and let (x, g) be a generic pair belonging to $\mathcal{Z}$. Then, the state x is $(\mathcal{Z}, \mathcal{Z}^{\text{new}}) - switchable$ iff

$$(x^{\text{new}}_{[i]}, g^{\text{new}}_{[i]}) \in \mathcal{Z}^{\text{new}}_{[i]}. \tag{16}$$

Proof: For the *sufficiency* of (16) please notice that it implies $(x, g) \in \mathcal{Z}^{\text{new}}_{[i]} \times \mathbb{R}^{(n-n^{\text{new}}_{[i]})\times(m-m^{\text{new}}_{[i]})}$. Hence, since $(x, g) \in \mathcal{Z}$, then $(x, g) \in \mathcal{Z} \cap (\mathcal{Z}^{\text{new}}_{[i]} \times \mathbb{R}^{(n-n^{\text{new}}_{[i]})\times(m-m^{\text{new}}_{[i]})})$ and, in turn, $(x, g) \in \mathcal{Z}^{\text{new}}$ because of (15). Concerning the *necessity* of (16), it is trivial to observe that *switchability* of x means $(x, g) \in \mathcal{Z} \cap \mathcal{Z}^{\text{new}}$ for some g. This observation directly implies (16) through definition (14) applied to $\mathcal{Z}^{\text{new}}$. ■

IV. Supervision Architecture

The novel supervision architecture, represented by the layered structure in Fig. 2, is introduced to address Problems 1 and 2. In this figure, each layer serves the layers below it and operates within the same communication network. The two lower layers, namely the *decentralized control level* and the *plant level*, represent the *physical* precompensated system Σ_i.

In this context, we focus on the design of the *cyber part* of the architecture shown in Fig. 2. Unlike the *physical* part, components within the *cyber* layer include signals from both the physical domain, such as those generated by sensors and those provided to actuators, and the cyber domain, such as messages exchanged between agents. Specifically, a *supervision* layer is responsible for implementing the local CG task, while a *topology* layer handles both the CR task and the OTD task.

It is important to note that within the cyber part, the two tasks in the *topology* layer are represented as connected by a horizontal link (see Fig. 2) because they require interagent communication

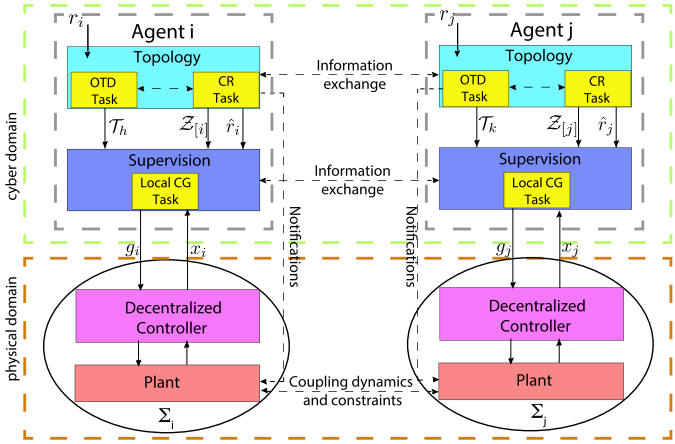

Fig. 2. Multilayer supervision architecture.

for a successful execution. In order to simplify the notation, we assume hereafter that the communication graph coincides with the constraints graph Γ.

Furthermore, while a new local CG task is started and completed at each time step, the task associated with the *topology* layer is assumed to run in the background and can be completed even after several time steps. For this reason, the local CG task is implemented using a noniterative procedure, while the tasks within the *topology* layer can execute iterative routines that return partial solutions at each iteration. These solutions are gradually refined before the task is completed.

A. Supervision Layer

The main goal of the coordinated action of all *supervision* levels of the agents is to select at each time instant, in a distributed way through the *local CG task*, an aggregated set-point $g(t)$ such that the pair $(x(t), g(t))$ belongs to $\mathcal{Z}$. This is achieved by resorting to the turn-based CG (TB-CG) noniterative noncooperative policy presented in [11], where the set $\mathcal{A}$ is assumed to be partitioned into q *turns* $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_q$ such that $\bigcup_{i=1}^{q} \mathcal{T}_i = \mathcal{A}$. At time t, only the agents belonging to a certain Turn, say $\mathcal{T}_h$ such that $h = t \mod q$, are allowed to update their local commands $g_i(t)$ while all others agents in $\mathcal{A} \setminus \mathcal{T}_h$ keep applying the same local command already used at time $t-1$. Within this policy, each agent in $\mathcal{T}_h$ receives from its neighbors the values of their states and of their previously applied commands, i.e., $x_{[i]}(t) = [x_i^T(t), \tilde{x}_i^T(t)]^T$ and $\tilde{g}_{[i]}(t-1)$. Then, the following condition:

$$(x_{[i]}(t), [g_i^T, \tilde{g}_i^T(t-1)]^T) \in \mathcal{Z}_{[i]} \tag{17}$$

can be used to characterize the class of all g_i (a slack variable here) that are feasible because the membership to $\mathcal{Z}$ is always satisfied. Finally, a unique suitable command $g_i(t)$ can be selected, e.g., by solving the optimization problem (18). This idea can be formalized as in Algorithm 1 where $\Psi_i = \Psi_i^T > 0 \; \forall i \in \mathcal{A}_i$.

Remark 3: In general, the communication graph could be sparser than Γ at a price of introducing a certain degree of conservativeness in solving Problem (17) ([26]) or robustifying the computation of the local sets $\mathcal{Z}_{[i]}$ ([27]) to decouple the structure of the constraints.

Algorithm 1: TB-CG - Agent i at time t [11].

INPUT: $\mathcal{T}_h, h \in \{1, \ldots, q\}$, $\mathcal{Z}_{[i]}$ and $\hat{r}_i(t)$ ▷ from the Topology Level
$\tilde{x}_i(t), \tilde{g}_i(t-1)$ ▷ from the neighbors
OUTPUT: $g_i(t)$

1: **if** $h = t \mod q$ **then**
2: solve

$$g_i(t) = \arg\min_{g_i} ||g_i - r_i(t)||^2_{\Psi_i} \tag{18}$$

subject to (17)

3: **else**
4: set $g_i(t) = g_i(t-1)$
5: **end if**
6: transmit $g_i(t)$ and $x_i(t)$ to the neighboring agents
7: return $g_i(t)$
8: set $t \leftarrow t+1$
9: go to 1

B. OTD Task Within the Topology Layer

This task is designed to deal with the online distributed reconfiguration of the existing turns sequence $\mathcal{S} := \{\mathcal{T}_k\}_{k=1}^{q}$ in response to a request of modification of the topology. The latter can arise whenever a generic *changing agent* $\bar{i}$, asking for modification of the set $\mathcal{Z}_{[\bar{i}]}$, needs to possibly modify its current neighborhood. Depending on the constraints introduced by $\mathcal{Z}_{[\bar{i}]}$, also some agents i belonging to $\mathcal{N}_{\bar{i}}$ may require modification of their sets $\mathcal{Z}_{[i]}$. More formally the following operations could be necessary.

- *Edges addition on agent* $i \in \mathcal{N}_{\bar{i}}$: $\mathcal{N}_i \leftarrow \mathcal{N}_i^{\text{new}} := \mathcal{N}_i \cup \{\text{some } j \in \mathcal{A} \setminus \mathcal{N}_i\}$.
- *Edges removal on agent* $i \in \mathcal{N}_{\bar{i}}$: $\mathcal{N}_{\bar{i}} \leftarrow \mathcal{N}_i^{\text{new}} := \mathcal{N}_i \setminus \{j \in \mathcal{N}_i\}$.

In this respect, it is worth recalling the general requirements to be guaranteed for a sequence of turns $\mathcal{S}$.

1) All agents of the network are periodically selected, i.e., $\cup_{i=1,..,q} \mathcal{T}_i = \mathcal{A}$..
2) In order to guarantee good tracking performance, the number of turns q should be as small as possible.

In [10], [11], it has been highlighted that determining a minimal and admissible sequence of turns that cover the entire $\Gamma(\mathcal{A}, \mathcal{E})$ is equivalent to solve the minimal vertex coloring problem ([31]) for the graph $\Gamma(\mathcal{A}, \mathcal{E})$.

In our case, such a problem cannot be solved in a centralized way as the OTD task works by means of neighbor-to-neighbor data exchange. On the other hand, to the best of our knowledge, no distributed algorithms are available for solving minimal vertex coloring problems [30]. Then, the OTD task can be only designed to guarantee that the number of used colors does not

Algorithm 2: OTD task - of agent $i \in \mathcal{N}_{\bar{i}}^{\text{new}}$ (edges addition).

asks for $\phi(j)$ to all $j \in \mathcal{N}_i^{\text{new}}$
find

$$p^\star = \arg\min_k \quad k$$
$$\text{s.t.} \quad p_k \in \mathcal{L}$$
$$p_k \neq \phi(j), \forall j \in \mathcal{N}_i^{\text{new}} \qquad (19)$$

if (19) has not solution **then**
  set $\mathcal{L} \leftarrow \mathcal{L} \cup \{p_{l+1}\}$
  notify new color set $\mathcal{L}$ to the neighbors
  set $\phi(\bar{i}) \leftarrow p_{l+1}$
else
  set $\phi(\bar{i}) \leftarrow p^\star$
end if

exceed the upper bound provided by the Brooks' theorem, that is $\Delta(\Gamma)+1$ ([31]).

In view of this, it is worth introducing an ordered set of colors $\mathcal{L} := \{p_1, p_2 \ldots, p_l\}$, such that $|\mathcal{L}| \leq \Delta(\Gamma)+1$ and the *coloring operator* $\phi : \mathcal{A} \to \mathcal{L}$. In the case of *edges addition*, the OTD task intervention can be mandatory as the current turns configuration $\mathcal{S}$ can result unfeasible. In this scenario, the *changing agent* $\bar{i}$ and/or its neighbors start seeking for the first available color in $\mathcal{L}$, by means of Algorithm 2 to be sequentially performed by each agent in $\mathcal{N}_{\bar{i}}^{\text{new}}$.

The latter algorithm enjoys the following property.

Proposition 1: Consider a graph $\Gamma(\mathcal{A}, \mathcal{E})$ whose vertices are properly colored by using the ordered color list $\mathcal{L} := \{p_1, p_2 \ldots, p_l\}, |\mathcal{L}| \leq \Delta(\Gamma)+1$, i.e., $(i,j) \in \mathcal{E} \Rightarrow \phi(i) \neq \phi(j)$, with $\phi(i), \phi(j) \in \mathcal{L}$, and a new graph $\Gamma'(\mathcal{A}, \mathcal{E}')$ where $\mathcal{E}' \leftarrow \mathcal{E} \cup \{(i', j)$, for some $j \in \mathcal{A} \setminus \mathcal{N}_{i'}$, for some $i'\}$. Then, by applying Algorithm 2 to node i', the number of colors used for coloring Γ' will not exceed $\Delta(\Gamma')+1$, i.e., $|\mathcal{L}| \leq \Delta(\Gamma')+1$.

Proof: Let us consider the worst case where $|\mathcal{L}| = \Delta(\Gamma)+1$. Two situations need to be analyzed as follows.

- $\Delta_{i'}(\Gamma') \leq \Delta(\Gamma)$. Then, $\mathcal{L}$ remains unchanged because there exists a $p_h \in \mathcal{L}$ such that $p_h \neq \phi(j), \forall j \in \mathcal{N}'_{\bar{i}}$.
- $\Delta_{i'}(\Gamma') > \Delta(\Gamma)$ that is $\Delta(\Gamma') = \Delta_{\bar{i}}(\Gamma')$. Then, in the worst case the list $\mathcal{L}$ is filled with at most $\Delta_{i'}(\Gamma') - \Delta(\Gamma)$ colors. Therefore, $|\mathcal{L}| = \Delta(\Gamma)+1+\Delta_{i'}(\Gamma') - \Delta(\Gamma) = \Delta_{i'}(\Gamma')+1 = \Delta(\Gamma')+1$. ■

Whenever an *edges removal* operation is performed, the current turns configuration $\mathcal{S}$ always remains feasible. However, a new turn configuration with a reduced number of colors could be determined to improve network control performance. This procedure can be designed according to the following steps.

1) Upon completion of the *edges removal* operation, the OTD tasks of the agents involved notify the other agents in Γ of the completion.
2) A distributed recoloring procedure is activated with a reduced number of colors.
3) When a new feasible color configuration is achieved, a new sequence of turns, denoted as $\mathcal{S}' := \mathcal{T}'^{q'}_{k\,k=1}, q' < q$, is used in Algorithm 1.

For the second step, the parallel iterative greedy algorithm presented in [29] can be used, which guarantees a correct coloring with at most $\Delta(\Gamma)+1$ colors. Since this algorithm may require several time steps to complete, it is designed to run in the background without affecting the activities at the TB *supervision* level. Further implementation details of this procedure are beyond the scope of this article. On the contrary, a crucial aspect of the proposed supervision architecture has been addressed: could the switching between two different turn configurations $\mathcal{S}$ and $\mathcal{S}'$ affect the feasibility of the scheme? The following result clarifies this potential issue.

Proposition 2: Let $\mathcal{T}_{t \bmod q} \in \mathcal{S}$ be the current active turn sequence at time $t > 0$ in Algorithm 1. Then, any transition from $\mathcal{T}_{t \bmod q}$ to any $\mathcal{T}'_{(t+1) \bmod q'} \in \mathcal{S}'$ is always safe with respect to the fulfillment of constraints (8).

Proof: The proof simply comes out by the turn Definition 9 and by the fact that in Algorithm 1 the previously applied command $g_i(t-1)$ is always feasible for all $i \in \mathcal{A}$. In fact, agents belonging to $\mathcal{T}'_{(t+1) \bmod q'}$ have not constraints conflict into the set $\mathcal{Z}$, then, at time $t+1$, they can safely update their commands while all agents not in $\mathcal{T}'_{(t+1) \bmod q'}$ keep applying their previously applied commands. Then, the overall constraints (8) are never violated. ■

Remark 4: Indeed, for sparse coupling graphs, q changes are infrequent and occur when the degree of a certain node exceeds the current degree of Γ. In this case, all agents in the network must be resynchronized. Anyway, Proposition 2 has been included to exclude possible feasibility issues.

C. CR Task Within the Topology Layer

This task is capable of determining, based on the conditions introduced in Section III-C, when the replacement of the set $\mathcal{Z}$ can occur during either Scenarios 1 or 2. It provides the *supervision* level with the desired reference and local CG parameters, i.e., the set $\mathcal{Z}_{[i]}$ in Algorithm 1, which may change in response to a request for modification of the set $\mathcal{Z}$.

Essentially, the CR task of a generic *changing agent* $\bar{i} \in \mathcal{A}$, requesting a modification of $\mathcal{Z}_{[\bar{i}]}$ at time $t = t_s$, deals with two possible situations depending on whether condition (16) is satisfied or not. In the affirmative, the *replacement* operation can be completed at time t_s. On the contrary, local states of $\Sigma_{\bar{i}}$ and its new neighbors need to be steered toward a *switchable* equilibrium $(x^{\text{new}}_{r^\star_{[\bar{i}]}})$. Finding such an equilibrium is quite challenging in a distributed context where agents follow noncooperative policies in optimizing their commands g_i.

Specifically, when no assumptions are made on the topology of Γ, there is no guarantee that a feasible *switchable* equilibrium can be found by involving only the neighbors of agent $\bar{i}$. The reason is that this procedure can be indirectly hindered by some nonneighboring agents coupled with neighbors of $\bar{i}$

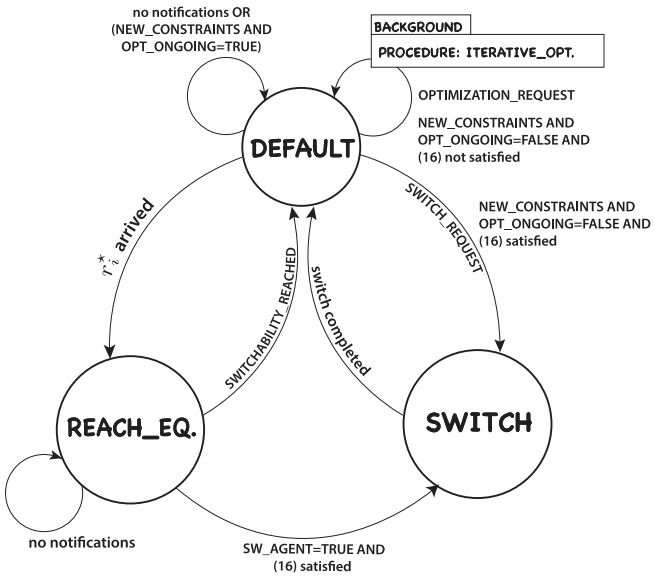

Fig. 3. CR task: Finite-state automata.

that apply certain references incompatible with the new OAS $\mathcal{Z}^{\text{new}}$. In the more general case, achieving a *switchable* global equilibrium $x_{r^\star}$ requires partial cooperation from all agents. They need to be instructed to track suitable references $r_i^\star$ for a certain amount of time, rather than their local nominal references $r_i(t)$.

Furthermore, the determination of $r_i^\star$ needs to be carried out based on local data only, allowing for the solution of the following multiobjective optimization problem:

$$\min_{g} \quad [||g_1 - g_1(t_s)||^2_{\Psi_1}, \ldots, ||g_N - g_N(t_s)||^2_{\Psi_N}] \tag{20}$$

$$\text{s.t.} \quad (x_g, g) \in \mathcal{Z} \sim \mathcal{B}_\delta \tag{21}$$

$$(x_{g_{[\bar{i}]}}, g_{[\bar{i}]}) \in \mathcal{Z}^{\text{new}}_{[\bar{i}]} \sim \mathcal{B}^{\bar{i}}_\delta \tag{22}$$

$$g = [g_1^T, \ldots, g_i^T, \ldots, g_N^T]^T.$$

A number of distributed approaches exist to compute a solution for the latter problem. Here, we adopt the procedure used in [33] and here referred as ITERATIVE_OPTIMIZATION. There, the agents compute their local feasible set-points in an iterative way by exploiting neighbors' information. Such an approach guarantees that a Pareto optimum $r^\star$ for (20)–(22) is reached in a finite time.

The entire CR task can be implemented by means of a finite-state automata consisting of three main logic modes: DEFAULT, SWITCH, REACH_EQ that are organized, as depicted in Fig. 3. The variable $s_i(t) \in \{\texttt{DEFAULT}, \texttt{SWITCH}, \texttt{REACH\_EQ}\}$ is used to store its current CR-task configuration while the boolean variables SW_AGENT and OPT_ONGOING are used as flags. In particular, the former denotes if the agent is a *changing agent* (agent who is requesting the switch) while the second is used to discriminate if the agent is involved in the ITERATIVE_OPTIMIZATION procedure to allow constraints modification. The three logic modes are hereafter detailed as follows.

1) DEFAULT Mode—Agent i at Time t: In this mode, when no switching are occurring, the *CR-task* basically conveys to the *supervision* level the reference $r_i(t)$. When the switching among two different constraint structures is ongoing, four possible notifications could be processed as follows.

i) NEW_CONSTRAINTS: this notification is conveyed by a not specified entity (perhaps the *user* supervising Σ_i plant) that wants to modify the current constraints configuration $\mathcal{Z}_{[i]}$ into $\mathcal{Z}^{\text{new}}_{[i]}$; in this case, if the internal flag OPT_ONGOING has the FALSE value, agent i set a flag SW_agent to TRUE value and gathers information from both its current and new (possible) neighborhoods. In this way, it can first check out if they are ready to replace $\mathcal{Z}_{[i]}$. Whenever the current pair $(x^{\text{new}}_{[i]}(t), g^{\text{new}}_{[i]}(t))$ satisfies condition (16), the *replacement* operation can be completed. In that case, it broadcasts a SWITCHABILITY_REACHED notification and the SWITCH mode becomes active. In the negative case, the ITERATIVE_OPTIMIZATION is activated for solving problem (20) after having notified a OPTIMIZATION_REQUEST to all its neighbors.

ii) OPTIMIZATION_REQUEST: this request can be received from a neighboring agent requiring that Algorithm 4 be activated in the background. Agents that activate such a procedure set the flag OPT_ONGOING to the TRUE value. When the ITERATIVE_OPTIMIZATION is completed, the new computed reference $r_i^\star$ is received in order for the subsystem state to approach the related equilibrium in REACH_EQ mode.

iii) SWITCH_REQUEST: this notification is received by a neighbor which has successfully checked the switchability conditions (16) so that the constraints can be safely modified by entering in the SWITCH mode.

2) SWITCH Mode—Agent i at Time t: In this mode Algorithm 2 in the OTD Task is activated from the *changing agent*. Once it is completed, all the operations required to finalize the switch can be adopted. In particular, the sets $\mathcal{Z}_{[i]}$ can be replaced $\leftarrow \mathcal{Z}^{\text{new}}_{[i]}$. Moreover, if a *dynamic structure modification* (Scenario 1) is ongoing a SWITCH_SUCCESS *notification* is sent to the (possible) *requesting user* that can properly update the parameters of the *physical* layer of each involved subsystem. Then, the *supervision* level of involved nodes in $\mathcal{N}_i^{\text{new}}$ starts working with sets $\mathcal{Z}^{\text{new}}_{[i]}$ while the original local references $\hat{r}_j \equiv r_j(t)$ for all $j \in \mathcal{A}$ can be restored in the local CG optimization problem (18).

3) REACH_EQ Mode: In this mode, the local *supervision* level is instructed to use the reference $\hat{r}_j \equiv r_j^\star$ in place of $r_j(t)$ in the optimization problem (18). This action is repeated until the SWITCHABILITY_REACHED is received so that the DEFAULT mode becomes active. In the case that agent i is the *changing agent*, the next active mode is SWITCH.

The above-mentioned logic is described in Algorithm 3.

Remark 5: It is worth remarking that the scheme is designed so as to satisfy the requests aroused from Scenarios 1 or 2 in the minimum time. For the sake of clarity, we do not consider the case of simultaneous requests although, in principle, the network could process more than one request at a time depending on

Algorithm 3: CR task - of agent i.

DEFAULT MODE
$s_i(t) \leftarrow$ DEFAULT
if NEW_CONSTRAINTS notification arrives AND OPT_ONGOING = FALSE **then**
  set SW_AGENT $\leftarrow$ TRUE
  transmit $\mathcal{Z}_{[i]}^{\text{new}}$ to $\mathcal{N}_i^{\text{new}}$ and $\mathcal{N}_i$
  receives $x_{[\bar{i}]}(t), x_{[i]}^{\text{new}}(t), g_{[i]}(t), g_{[i]}^{\text{new}}(t)$
  if $(x_{[i]}^{\text{new}}(t), g_{[i]}^{\text{new}}(t))$ satisfies (16) **then**
  notify SWITCHABILITY_REACHED to $\mathcal{N}_i$ and $\mathcal{N}_i^{\text{new}}$ and go to SWITCH MODE
  else
  notify OPTIMIZATION_REQUEST to $\mathcal{N}_i$
  activate ITERATIVE_OPT. in background
 end if
end if
if SWITCH_REQUEST notification arrives from $j \in \mathcal{N}_i$ **then**
  receives $\mathcal{Z}_{[i]}^{\text{new}}$
  go to SWITCH MODE
 end if
 if OPTIMIZATION_REQUEST notification arrives from some $j \in \mathcal{N}_i$ **then**
  OPT_ONGOING$\leftarrow$ TRUE
  notify OPTIMIZATION_REQUEST to $\mathcal{N}_i \setminus \{j\}$
  activate ITERATIVE_OPT. in background
 end if
if $r_i^\star$ is received from ITERATIVE_OPT. instance **then**
  OPT_ONGOING$\leftarrow$ FALSE
  go to REACH_EQ MODE
 end if
convey $\hat{r}_i(t) = r_i(t)$
$t \leftarrow t+1$
go to 1
SWITCH MODE
$s_i(t) \leftarrow$SWITCH
if SW_AGENT=TRUE **then**
  activate OTD Task and wait for com-
  pletion of possible addition/removal edges operation
  notify SWITCH_REQUEST to $\mathcal{N}_i$ and $\mathcal{N}_i^{\text{new}}$
  send $\mathcal{Z}_{[j]}^{\text{new}}$ to each $j \in \mathcal{N}_i$
  SW_AGENT$\leftarrow$ FALSE
  notify SWITCH_SUCCESS to the requesting user ▷ not mandatory
 end if
set $\mathcal{Z}_{[i]} \leftarrow \mathcal{Z}_{[i]}^{\text{new}}$
go to DEFAULT MODE
REACH_EQ MODE
$s_i(t) \leftarrow$REACH_EQ
convey $\hat{r}_i(t) \leftarrow r_i^\star$
if SW_AGENT=TRUE **then**
  receives $x_{[\bar{i}]}(t), x_{[i]}^{\text{new}}(t), g_{[i]}(t), g_{[i]}^{\text{new}}(t)$
  if $(x_{[i]}^{\text{new}}(t), g_{[i]}^{\text{new}}(t))$ satisfies (16) **then**
  notify SWITCHABILITY_REACHED to $\mathcal{N}_i$ and $\mathcal{N}_i^{\text{new}}$ and go to SWITCH MODE
  end if
 else
  if SWITCHABILITY_REACHED received **then**
  go to DEFAULT MODE
  end if
end if
$t+1 \leftarrow t$
go to 39

the existing couplings. Sparse networks can easily deal with multiple requests, while if coupled subsystems receive separate requests at the same time, a specific prioritization protocol can be adopted to process the requests in a sequential way. In the worst case (full connected topology), such an approach induces the entire network to process one request at a time.

Remark 6: An interesting aspect of Algorithm 3 deserves attention and is related to how much time agents spend in the REACH_EQ mode during switching operations. In the worst case, in order to reach a switchable state, all other agents need to temporarily track a virtual reference r^*. In certain cases, this would negatively influence or interrupt the normal operation of these agents. However, it would be worth establishing what is more negative for the entire network: i) to deny a constraint topology modification (perhaps mandatory or anyway useful to increase global performance)? ii) to allow such a modification at the price of interrupting normal operations of some agents? The answer is not trivial because it strongly depends on the type of application. For instance, for sparse networks the influence of such a procedure is low for not directly involved agents (please refer to the first example in Section VI.). For dynamically coupled subsystems (power grids, water networks), a structural modification could indeed influence the operations of the entire network. Anyway, such operations are not frequent in this type of network and are often performed when they become no longer postponable.

V. Properties

This section presents some interesting properties of the abovementioned supervision architecture. For the forthcoming analysis, we need to introduce the following statement holding true for generic linear systems (3) satisfying Assumption (**A1**).

P1. Let $g(t) \equiv g, \forall t$ being applied to system Σ in (8). Then, $\forall \lambda > 0$ there exists a real $\beta(\lambda) > 0$ such that $\|x(0) - x_g\| \leq \beta(\lambda) \Rightarrow \|\hat{x}(t) - x_g\| \leq \lambda, \forall t \geq 0, \forall g \in \mathbb{R}^m$.

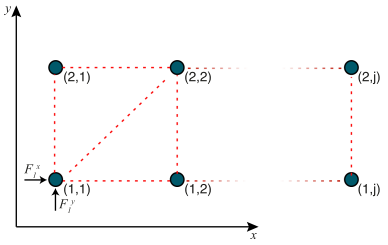

Fig. 4. Planar moving objects. The red dashed edges indicates presence of coupling constraints between two ASVs according to (29).

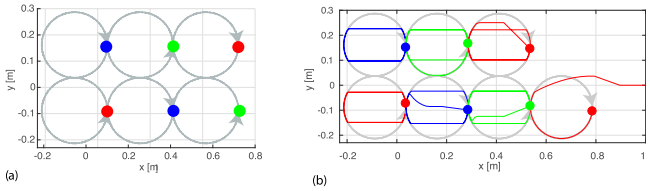

Fig. 5. Moving objects(colored spots) with desired references (gray lines): Left before *plug-in* operaton, *right:* After plug-in operation.

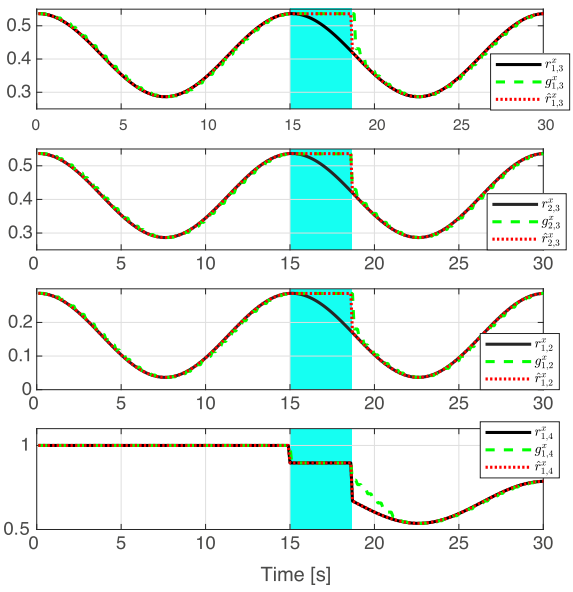

Fig. 6. Applied commands on x-axis related to some agents in the network.

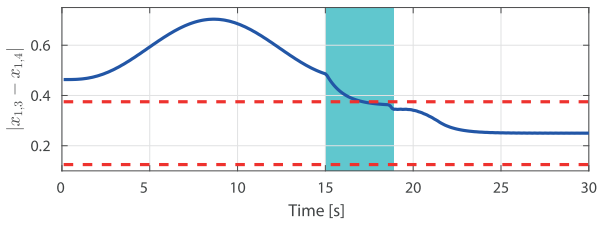

Fig. 7. Constrained variable between agent $\{1,3\}$ and $\{1,4\}$.

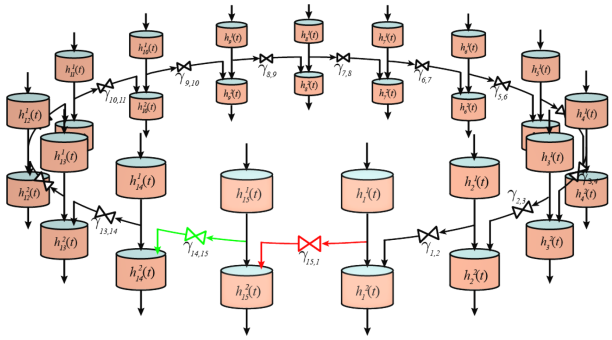

Fig. 8. Fifteen cascaded two-tank system.

A first result about the finite time transitions among two replaceable sets $\mathcal{Z}$ and $\mathcal{Z}^{\text{new}}$ is given by the following Lemma.

Lemma 1: Consider subsystems (2) along with the distributed supervision architecture depicted in Fig. 2. Then, any transition among two *replaceable* sets $\mathcal{Z}$ and $\mathcal{Z}^{\text{new}}$, required by an agent $\bar{i}$ at time $t = t_s^{\text{start}}$, is performed in a finite time under the assumption that $\text{Proj}_g\{\mathcal{Z}\}$ is viable according to Definition 4, $s_i(t)$=DEFAULT, $\forall i \in \mathcal{A}$ and a pair $(x(t), g(t)) \in \mathcal{Z}$ exists. This implies that there exists $t_s^{\text{end}} \in [t_s^{\text{start}}, \infty)$ at which the operation $\mathcal{Z}_{[\bar{i}]} \leftarrow \mathcal{Z}_{[\bar{i}]}^{\text{new}}$ (row 37 in Algorithm 3) is executed by the CR tasks of agent $\bar{i}$ and agents in $\mathcal{N}_{\bar{i}}$ and $\mathcal{N}_{\bar{i}}^{\text{new}}$.

Proof: Let us consider the worst case where $x(t_s^{\text{start}})$ is not $(\mathcal{Z}, \mathcal{Z}^{\text{new}})$-*switchable*. In this case, as described in Algorithm 3, the CR task of each agent activates the ITERATIVE_OPTIMIZATION procedure of [33]. Therefore, all agents are able to find in a finite time, say it τ, a *switchable* equilibrium $(x_{r^\star}, r^\star)$ solution of (20)–(22). Then, the CR-task of each agent starts working in REACH_EQ mode. Please notice that thanks to condition (15) and (22), one has

$$[x_{r^\star}^T, (r^\star)^T]^T \in \text{In}\{\mathcal{Z} \cap \mathcal{Z}^{\text{new}}\}. \tag{23}$$

By resorting to Theorem 1 of [11], it is guaranteed that by applying Algorithm 1 with $\hat{r}_i(t) \equiv r_i^\star$ in place of $r_i(t)$, the signals $g_i(t), \forall i \in \mathcal{A}$ asymptotically converges to $r_i^\star, \forall i \in \mathcal{A}$. As a consequence, thanks to property **P1** and (23), we can state that there exists an arbitrarily small $\epsilon > 0$ such that

$$(x_{r^\star} \oplus \mathcal{B}_\epsilon, (r^\star)) \subset \mathcal{Z} \cap \mathcal{Z}^{\text{new}} \tag{24}$$

and a related $t' > t_s^{\text{start}} + \tau$ such that

$$\|x(t) - x_{r^\star}^T\| \leq \epsilon \ \forall t \geq t'. \tag{25}$$

Hence, the *switchability* condition (16) is successfully checked at time $t_s^{\text{end}} = t'$ by agent $\bar{i}$ (row 43 of Algorithm 3). Then, the switching operation $\mathcal{Z}_{[\bar{i}]} \leftarrow \mathcal{Z}_{[\bar{i}]}^{\text{new}}$ can be safely carried out by all agents $i \in \mathcal{N}_{\bar{i}}$ that successively enters into the DEFAULT mode. ■

Finally, the main properties of the above-mentioned supervision architecture can be summarized in the following Theorem.

Theorem 2: Consider subsystems (2) along with the distributed supervision architecture depicted in Fig. 2, and assume that all agents in $\mathcal{A}$, grouped in turns $\mathcal{T}_i$, periodically perform Algorithm 1. Then:

1) (*feasibility*) the overall system (3) acted by agents implementing Algorithm 1 under the inputs of local *topology* layer never violates the constraints (8) also when a replacement $\mathcal{Z} \leftarrow \mathcal{Z}^{\rm new}$ with $(\mathcal{Z}, \mathcal{Z}^{\rm new})$ being a *replaceable* pair;
2) (*stability*) for any sequence $\Pi := \{\mathcal{Z}^k\}_{k=0}^{v-1}$ of OASs to be consecutively replaced to $\mathcal{Z}$ because of **Scenarios** 1 or 2 requests, the state $x(t)$ remains confined into the set $\Xi := \bigcup_{k=0}^{v} \mathcal{X}^k$, with $\mathcal{X}^k := {\rm Proj}_x\{\mathcal{Z}^k\}$ and $v > 0$ a not specified large integer. Moreover Ξ is a compact set if the pair (A, E_c), with $A := \left[\begin{smallmatrix} \overset{\Phi}{0_{m\times n},} & \overset{G}{I_{n\times m}} \end{smallmatrix}\right]$ and $E_c := [H_c, L]$, remains completely observable in spite of all possible replacements occurring on Φ;
3) (*convergence*) for any finite sequence $\Pi := \{\mathcal{Z}^k\}_{k=0}^{v-1}$ of OASs to be consecutively replaced to $\mathcal{Z}$, under the assumption that ${\rm Proj}_g\{\mathcal{Z}^k\}$ is viable for all $k \in [0, v-1]$ (see Definition 4) and $r(t) \equiv [r_1^T, \ldots, r_N^T]^T, \forall t$ is a constant set-point, the sequence of solutions $g(t) = [g_1^T(t), \ldots, g_N^T(t)]^T$ asymptotically converges to a PO stationary (constant) solution of the following multiobjective optimization problem:

$$\min_g [\| g_1 - r_1 \|_{\Psi_1}^2, \ldots, \| g_i - r_i \|_{\Psi_2}^2, \ldots, \| g_N - r_N \|_{\Psi_N}^2]$$
$$(x_g, g) \in \mathcal{Z} \equiv \mathcal{Z}^{v-1}. \tag{26}$$

The solution is given by either r, whenever $(x_r, r) \in \mathcal{Z}$, or by any other PO solution $\hat{r}$ such that $(x_{\hat{r}}, \hat{r}) \in \mathcal{Z}$ otherwise. Moreover

$$\lim_{t\to\infty} y(t) = \hat{r}. \tag{27}$$

Proof:

1) In normal conditions, when no constraint modifications are required, the CR task of each agent works in DEFAULT mode. Therefore, the TB-CG policy is implemented trough Algorithm 1 and the constraints are never violated by construction of the set $\mathcal{Z}$ in (9). Even in the case where a replacement operation between $\mathcal{Z}$ and $\mathcal{Z}^{\rm new}$ is started at a certain time $t = t_s^{\rm start}$ and completed at time $t = t_s^{\rm end}$, Algorithm 1 running under REACH_EQ mode, guarantees $(x(t), g(t)) \in \mathcal{Z}, \forall t \in [t_s^{\rm start}, t_s^{\rm end})$. Finally, when the replacement $\mathcal{Z} \leftarrow \mathcal{Z}^{\rm new}$ occurs at time $t = t_s^{\rm end}$, $(x(t_s^{\rm end}), g(t_s^{\rm end}))$ is a $(\mathcal{Z}, \mathcal{Z}^{\rm new})$-*switchable* pair, hence, thanks to Theorem 1, $(x(t), g(t))) \in \mathcal{Z}^{\rm new}, \forall t \geq t_s^{\rm end}$. Then, the constraints (8) are never violated even with the new set $\mathcal{Z}^{\rm new}$. Also in the case that a different turn configuration is adopted, constraints satisfaction is guaranteed by Proposition 2.
2) From previous item, it directly follows that $(x(t), g(t))$ is confined into $\bigcup_{k=0}^{v} \mathcal{Z}^k$ and, in turn, $x(t) \in \Xi, \forall t > 0$. To prove that the Ξ set is compact it is enough to prove that a generic $\mathcal{Z}^k$ is compact. Because (A, E_c) is an observable pair, the matrix $\Theta^T\Theta$ is nonsingular, where $\Theta := [E_c^T, (E_cA)^T, \ldots, (E_cA^{n+m-1})^T]^T$. It follows that the linear map between (x, g) and c admits the left inverse operator $[x^T, g^T]^T = (\Theta^T\Theta)^{-1}\Theta^T[c^T(0, x, g), \ldots, c^T(n+m-1, x, g)]^T$.
3) When the CR-task of each agent works in DEFAULT mode the proposed scheme is equivalent to the TB-CG scheme of [11] that has been proved to converge to a PO for problem (26) whenever $r(t)$ is a constant set-point. Hence, condition (27) directly follows from Assumptions **A.1**–**A.2**. In view of this, to prove this latter statement for our purposes, it is enough to prove that, thanks to Lemma 1, each OAS replacement $\mathcal{Z}^k \leftarrow \mathcal{Z}^{k+1}$, $\mathcal{Z}^k, \mathcal{Z}^{k+1} \in \Pi$ is completed in a finite time interval. As a consequence, there exists a future finite time instant after which all agents permanently will enter in the DEFAULT mode. ■

VI. Illustrative Examples

A. Decoupled Systems

Let us consider a set of N decoupled objects moving in a 2-D space and depicted in Fig. 4. These objects can be seen as autonomous surface vehicles (ASVs) that are devoted to the patrolling of a certain water surface. The monitoring of extensive water resources ([34], [35]) is usually performed for either security or environmental reasons. In our specific application, these vehicles are instructed to monitor some nonoverlapping areas according to certain prespecified trajectories. Such a task poses significant technical issues related to collision avoidance and proximity constraints that require an effective coordination among the vehicles.

To this end, in this section, a CPS is designed according to the supervision architecture of Fig. 2 involving such vehicles. More in details, the physical part concerning the generic object (i, j) depicted in Fig. 4 can be modeled as

$$\begin{aligned} m\ddot{x}_{i,j} &= F_{i,j}^x \\ m\ddot{y}_{i,j} &= F_{i,j}^y \end{aligned} \tag{28}$$

where $(x_{i,j}, y_{i,j})$ are the cartesian coordinates of the (i, j)th object position w.r.t a fixed Cartesian reference frame and $(F_{i,j}^x, F_{i,j}^y)$, the components along the same reference frame of the forces acting as inputs on the subsystems. The value $m = 1$[Kg] will be assumed in the simulations. In order to design the cyber part of this CPS, the models (28) have been discretized with a sampling time of $T_c = 0.1$ $[s]$ and an optimal LQ state-feedback local controller is used as a precompensator for each object.

All objects are subject to the following local and coordination coupling constraints:

$$\begin{cases} |F_{i,j}^p(t)| \leq 2\ [N] p = x, y \\ 0.125[\text{m}] \leq |x_{i,j+1}(t) - x_{i,j}(t)| \leq 0.375[\text{m}] \\ 0.125[\text{m}] \leq |x_{i,j}(t) - x_{i,j-1}(t)| \leq 0.375[\text{m}] \\ 0.125[\text{m}] \leq |y_{i+1,j}(t) - y_{i,j}(t)| \leq 0.375[\text{m}] \\ 0.125[\text{m}] \leq |y_{i,j}(t) - y_{i-1,j}(t)| \leq 0.375[\text{m}] \\ 0.125[\text{m}] \leq |x_{i+1,j+1}(t) - x_{i,j}(t)| \leq 0.375[\text{m}] \\ 0.125[\text{m}] \leq |y_{i+1,j+1}(t) - y_{i,j}(t)| \leq 0.375[\text{m}] \\ 0.125[\text{m}] \leq |x_{i,j}(t) - x_{i-1,j-1}(t)| \leq 0.375[\text{m}] \\ 0.125[\text{m}] \leq |y_{i,j}(t) - y_{i-1,j-1}(t)| \leq 0.375[\text{m}] \\ \forall t \in \mathbb{Z}_+ \end{cases} \tag{29}$$

that can be modeled by means of the graph depicted in Fig. 4, where each object represents a vertex while the red dashed edges denote existence of constraints between certain pairs of assets. In this case, agents are organized into three Turns identified by green, blue, and red color, respectively (see Fig. 5). Moreover, each vehicle has been instructed to track a "circular" reference $r_{i,j}(t)$ depicted in gray line in Fig. 5.

Numerical experiments have been accomplished simulating the following scenario, involving initially six ASVs, depicted in Fig. 5(a).

Until time instant $t = 14.9[s]$ the vehicles evolve by tracking their local nominal references $r_{i,j}(t)$, at time $t = 15[s]$ a new agent, say it $\{1, 4\}$ asks for a plug-in operation to agent $\{1, 3\}$. Then, the involved agents should reconfigure their local CGs to satisfy this request by getting the new configuration depicted in Fig. 5(b).

The numerical results are reported in Figs. 6 and 7. In particular, it is possible to observe that until time instant $t = 14.9[s]$ the agents in the network are able to track their local nominal references by performing Algorithm 1. Agent $\{1, 4\}$, which is not yet joining the network, applies a constant reference $g_{1,3}(t) \equiv [1, 0]^T, t = [0, 14.9]$. At time $t = 15[s]$, when the *plug-in* request is raised, the states of agents $\{1, 3\}$ and $\{1, 4\}$ are not *switchable*. For this reason, the *CR task* of all agents determine a feasible references related to a suitable *switchable* equilibria to be reached in finite time trough Algorithm 4. In particular, agents $\{1, 3\}$ and $\{1, 4\}$ start executing Algorithm 1 in REACH_EQ mode, by applying the new references $r^\star_{1,3} = [0.537, -0.083]^T$ and $r^\star_{1,4} = [0.896, 0.0]^T$, respectively. In this way, 36 time steps are required to reach a *pluggable* state. In fact, at time instant $t = 18.6$[s], as depicted in Fig. 7, the coupling constraints between agents $\{1, 3\}$ and $\{1, 4\}$ are strictly fulfilled and the *plug-in* operation can safely be completed. In particular, the *OTD task* of agent $\{1, 4\}$, trough Algorithm 2, assigns the "red" color to the new joined Agent $\{1, 4\}$ so that it can participate in the round-robin command updating policy of the *supervision* level.

B. Coupled Systems

A second CPS example considers as physical part the water tank system depicted in Fig. 8 and composed by fifteen interconnected cascaded two-tank subsystems. Such a plant is a large-scale generalization of a well-known four-tank system used as benchmark for validating several control strategies in either simulation ([36]) or experimental ([37], [38]) scenarios.

More in details, each two-tank subsystem in Fig. 8 is modeled by the following linearized equations:

$$\Sigma_i := \begin{cases} \dot{h}_i^1 = -\alpha h_i^1 + \rho u_i \\ \dot{h}_i^2 = -\alpha h_i^2 + \sigma_i(\gamma_{i,j})\alpha h_i^1 + \beta_{i,j}(\gamma_{i,j}) h_j^1 \end{cases}$$

where, for each $q = 1, 2$, $h_i^q[m]$ are the liquid levels in the tanks and $u_i[m^3/s]$ are the liquid flows supplied by the pumps. The split between the upper and lower tanks is modeled by coefficients σ_i, $\beta_{i,j}$, $i \in \mathcal{A}$, $j = (i + 1) \bmod 15$, that depend on the state "CLOSED/OPEN" of certain valves and that can be remotely accessed even by the *topology layer*. Their behavior is codified by means of the variables $\gamma_{i,j} : \mathbb{R} \to \{0, 1\}$ in the following way:

$$\Sigma_i : \begin{cases} \sigma_i(\gamma_{i,j}) = \sigma_i(\gamma_{i,j}) = (1 - \gamma_{i,j}/2) \\ \beta_{i,j}(\gamma_{i,j}) = \beta_{i,j}(\gamma_{i,j}) = \gamma_{i,j}/2. \end{cases}$$

Moreover $\rho = 1[1/\mathrm{m}^2]$ and $\alpha = 0.025[1/\mathrm{s}]$.

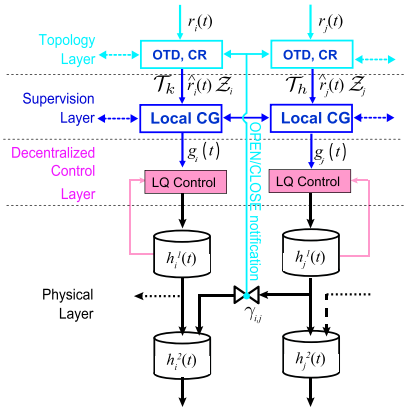

Fig. 9. Multilayer supervision architecture of Fig. 2 applied to agents in Fig. 8.

As described in Fig. 9, each cascaded two-tank subsystem is supervised by an *agent* (cyber layer) with the aim of regulating the levels $h_i^1(t)$, $i \in \mathcal{A}$ according to the supervision architecture proposed in Section V. Local decentralized tracking LQ output feedback controllers are implemented for managing the incoming water flows $u_i(t)$.

The system is discretized with sampling time $T_c = 0.5$ s. The following local pointwise-in-time set-membership constraints are assumed to be enforced

$$0.1[\mathrm{m}] \leq h_i^1 \leq 0.5[\mathrm{m}], 0.22[\mathrm{m}] \leq h_i^2 \leq 0.37[\mathrm{m}], \forall i \in \mathcal{A}$$
$$0 \leq u_i \leq 3[\mathrm{cm}^3/\mathrm{s}], \forall i \in \mathcal{A}. \quad (30)$$

Please notice that such constraints, although local in principle, give rise to an OAS $\mathcal{Z}$ with global conflicts because of possible dynamic interconnections among the subsystems. As a consequence, its structure can vary according to the values of variables $\gamma_{i,j}$.

In the numerical simulations, subsystems aim at maintaining the level of its upstream tank close to the following signals $r_i(t) = 0.32[\mathrm{m}]$, $i \in [1, 13]$, $r_{14}(t) = 0.1[\mathrm{m}]$, $r_{15}(t) = 0.48[\mathrm{m}]$, $\forall t \geq 0$ without violating constraints (30). Furthermore, during the simulation, it is required that the configuration of the valves vary according to the following scenario.

Until time instant $t = 14.9[s]$ $\gamma_{i,j}(t) = 1$, $i \in [1, 14]$, $j \in [2, 15]$, and $\gamma_{15,1}(t) = 0$, the network maintains the path-shaped constraints topology described in Fig. 10(a). At time $t = 150[s]$, Agent 15 receives a request to open the valve $\gamma_{15,1}$, (red pipe in Fig. 8) in order to achieve the constraints topology depicted in Fig. 10(b). Finally at time $t = 299.5[s]$, Agent 15 is asked to open valves $\gamma_{14,15}$ (green pipe in Fig. 8) and $\gamma_{15,1}$ in order to perform a plug-out operation and achieve the constraints topology depicted in Fig. 10(c).

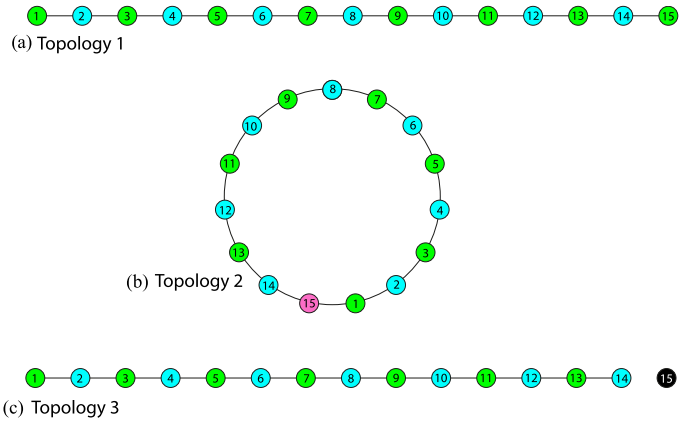

Fig. 10. Some possible topologies for the graph Γ of the water network based on the values of valves state γ. (a) Path topology. (b) Ring topology. (c) Path topology with agent 15 in stand-alone configuration.

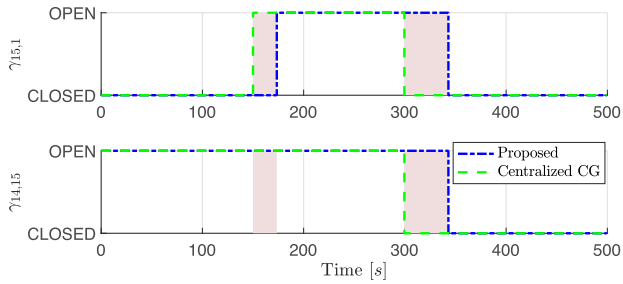

Fig. 11. State of valves being modified during the simulation.

The simulations have been carried out on MATLAB running on an Apple M1 MacBook Pro and have involved a comparison with the traditional CG scheme of [18] here implemented as a centralized supervisor able to manage all the set-points $g_i(t)$ at the same time. Since such a strategy is not provided with *topology* layer, it is no able to take into account possible changes in the constraints topology. For this reason, in these simulations the centralized CG has been designed according to the constraints topology Fig. 10(a).

All simulations results have been collected in Figs. 11–14. There, the behavior of directly involved agents (1, 14, 15) in constraints modifications have been shown along with that pertaining to Agent 7, whose dynamics is not affected by parameters changes along the simulations. When the above-mentioned water network is supervised by the centralized CG, the envisaged topology modification suddenly occurs (see Fig. 11) without any reconfiguration. Within this context, constraints are first violated on the downstream tank of $\Sigma_1 5$ in the time range $[150, 299.5]$[s], because the new in-flowing water coming from the upstream tank of Σ_1 is not properly accounted for. Second, after the sudden *plug-out* of Σ_5, Agent 14 is no more able to satisfy constraints on its downstream tank at time $t = 315.5$[s], as it is not aware that the inflow related to Σ_{15} is no more provided. On the contrary, a different behavior of the water network arises when the proposed supervision architecture is implemented. In fact, at time $t = 150$[s], when Agent 15

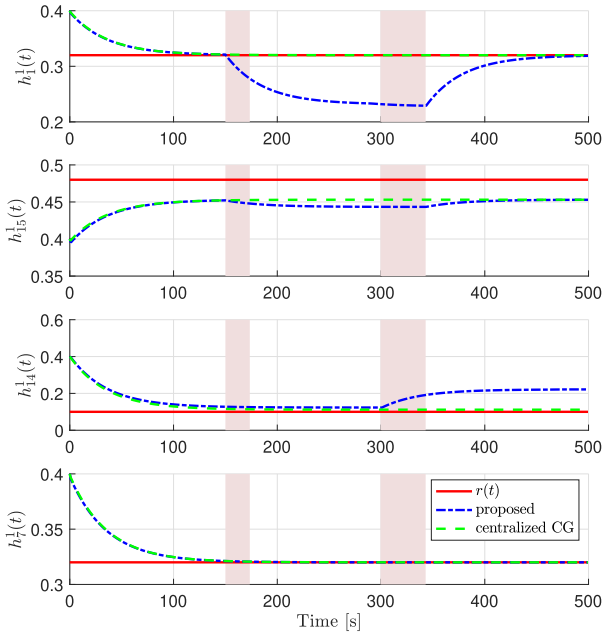

Fig. 12. Water level in the upperstream tanks.

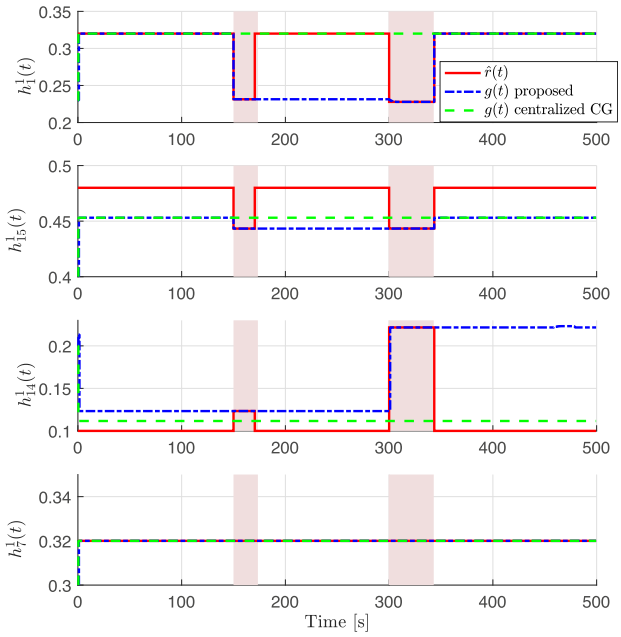

Fig. 13. References for the upperstream tanks.

request is raised, the current state $x(150)$ is still not *switchable* because the current pair $(x^{\text{new}}_{[15]}(150), g^{\text{new}}_{[15]}(150)), x^{\text{new}}_{[15]}(150) = [h_1(150),\ h_{14}(150),\ h_{15}(150)]^T, g^{\text{new}}_{[15]}(150) = [g_1(150),\ g_{14}(150), g_{15}(150)]^T$ does not satisfy condition (16). Then, the agents, by means of the `ITERATIVE_OPTIMIZATION` procedure compute in almost 0.34[s] the new feasible reference $r^\star$ to be tracked, that is $r_i^\star = 0.23\, i = 1, 2, r_i^\star = 0.32\, i \in [3, 12]$, $r_{13}^\star = 0.3213$, $r_{14}^\star = 0.12$, $r_{15}^\star = 0.44$ (see the shady zoomed areas in Fig. 13). In the meanwhile, the *OTD* task of Agent 2 determines a new turns configuration by performing the

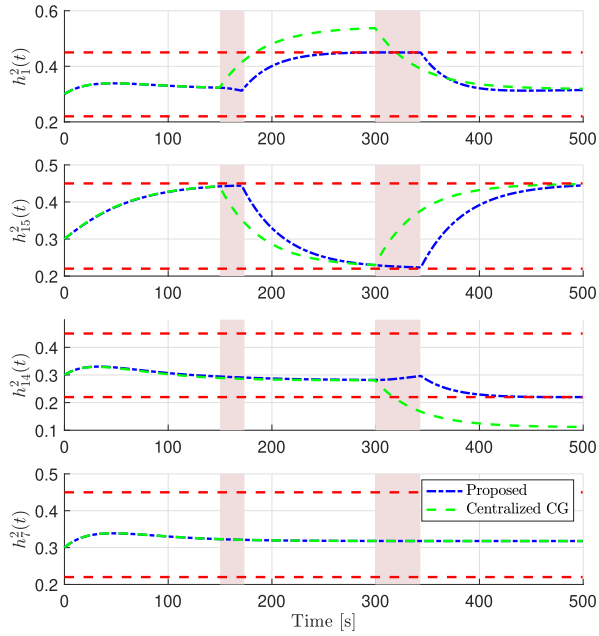

Fig. 14. Water level in the downstream tanks.

TABLE I
AVERAGE CPU TIME [MS]

Proposed Architecture (Agent 15)			Centralized CG
Topology 1	Topology 2	Topology 3	
0.68	0.69	0.67	8.9

graph coloring procedure stated in Algorithm 2. In this way, the *supervision* layer of each agent executes Algorithm 1 in REACH_EQ mode with $\hat{r_i}(t) = r^\star$ for 47 time steps until $t = 173.5$[s] when $(x^{\text{new}}_{[15]}(173.5), g^{\text{new}}_{[15]}(173.5))$ satisfies (16) and valve $\gamma_{15,1}$ can be opened (see first shady area in Fig. 11). Within the new topology both Σ_15 and Σ_1 need to decrease the level of their upstream tanks in order to fulfill the constraints.

A similar reasoning applies at time $t = 299.5$[s] when Agent 15 asks to *plug-out*. In this case, the network reaches a new *switchable* state after 87 time steps ($t = 343.5$[s]). Please notice in Fig. 13 that after the *plug-out* operation Agent 1 is again able to track its desired reference while Agent 14 is required to increase its upstream level for compensating the missing inflow of Σ_{15}. Along the simulation, it is interesting to observe the behavior of Agent 7 (see Figs. 12–14) testifying that the above described operations actually do not affect agents that are sufficiently *far* in the constraints graph from those neighborhoods subject to modifications.

A further aspect analyzed in this example concerns the computational scalability of the proposed scheme. In this respect, Table 1 compares the average CPU time per step required to Agent 15 for computing its reference $g_i(t)$ with that related to the centralized CG. From such a table it is possible to observe that the computational burden for Agent 15 is almost the same regardless the number of its neighbors. On the contrary, in a centralized configuration, the CPU time increases (almost linearly in this case the) with the number of involved subsystems.

VII. Conclusion

In this work, distributed CG theory is extended to address the reference management problem in a network of dynamically coupled subsystems with time-varying coordination constraints and dynamic coupling.

To achieve this, a novel multilevel supervision architecture is proposed to coordinate the behavior of each agent in the network. At the lower level, this strategy implements a TB distributed CG algorithm based on data provided by the higher level. The higher level is responsible for instructing the reconfiguration of local CGs in a distributed manner and determining the composition of turns in response to topology variation requests.

Simulation results performed on both decoupled and coupled CPSs demonstrate the advantages and flexibility of the proposed supervision framework.

Future research will focus on investigating the presented strategy in the case of quantized and rate-limited communication between agents, even in the presence of packet loss.

References

[1] J. A. Fax and R. M. Murray, "Information flow and cooperative control of vehicle formations," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1465–1476, Sep. 2004.
[2] A. Jadbabaie, J. Lin, and A. S. Morse, "Coordination of groups of mobile autonomous agents using nearest neighbor rules," *IEEE Trans. Autom. Control*, vol. 48, no. 6, pp. 988–1001, Jun. 2003.
[3] Z. Lin, M. Broucke, and B. Francis, "Local control strategies for groups of mobile autonomous agents," *IEEE Trans. Autom. Control*, vol. 49, no. 4, pp. 622–629, Apr. 2004.
[4] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time delays," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1520–1533, Sep. 2004.
[5] H. G. Tanner, G. J. Pappas, and V. Kumar, "Leader-to-formation stability," *IEEE Trans. Robot. Automat.*, vol. 20, no. 3, pp. 443–455, Jun. 2004.
[6] J. Shi, J. Wan, H. Yan, and H. Suo, "A survey of cyber-physical systems," in *Proc. Int. Conf. Wireless Commun. Signal Process.*, 2011, pp. 1–6.
[7] A. Platzer, *Logical Foundations of Cyber-Physical Systems*, vol. 662. Cham, Switzerland: Springer, 2018.
[8] F. M. Zhang, K. Szwaykowska, W. Wolf, and V. Mooney, "Task scheduling for control oriented requirements for cyber-physical systems," in *Proc. Real-Time Syst. Symp.*, 2008, pp. 47–56.
[9] M. W. Maier, "Architecting principles for systems-of-systems," *Syst. Eng.: J. Int. Council Syst. Eng.*, vol. 1, no. 4, pp. 267–284, 1998.
[10] Y. Kuwata, A. Richards, T. Schouwenaars, and J. P. How, "Decentralized robust receding horizon control for multi-vehicle guidance," in *Proc. Amer. Control Conf.*, 2006, pp. 2047–2052.
[11] A. Casavola, F. Tedesco, and E. Garone, "A distributed command governor based on graph colorability theory," *Int. J. Robust Nonlinear Control*, vol. 28, no. 8, pp. 3056–3072, 2018.
[12] P. Trodden and A. Richards, "Adaptive cooperation in robust distributed model predictive control," in *Proc. IEEE Control Appl., Intell. Control*, 2009, pp. 896–901.
[13] A. Núñez, C. Ocampo-Martínez, B. D Schutter, F. Valencia, J. López, and J. Espinosa, "A multiobjective-based switching topology for hierarchical model predictive control applied to a hydro-power valley," *IFAC Proc. Volumes*, vol. 46, no. 20, pp. 529–534, 2013.
[14] F. Fele, J. M. Maestre, S. M. Hashemy, D. M. d. l. Peña, and E. F. Camacho, "Coalitional model predictive control of an irrigation canal," *J. Process Control*, vol. 24, no. 4, pp. 314–325, 2014.
[15] S. Riverso, M. Farina, and G. Ferrari-Trecate, "Plug-and-play decentralized model predictive control for linear systems," *IEEE Trans. Autom. Control*, vol. 58, no. 10, pp. 2608–2614, Oct. 2013.

[16] S. Lucia, M. Kögel, and R. Findeisen, "Contract-based predictive control of distributed systems with plug and play capabilities," *IFAC-PapersOnLine*, vol. 48, pp. 205–211, 2015.
[17] E. F. Asadi and A. Richards, "Scalable distributed model predictive control for constrained systems," *Automatica*, vol. 93, pp. 407–414, 2018.
[18] A. Bemporad, A. Casavola, and E. Mosca, "Nonlinear control of constrained linear systems via predictive reference management," *IEEE Trans. Autom. Control*, vol. 42, no. 3, pp. 340–349, Mar. 1997.
[19] E. Garone, S. D. Cairano, and I. Kolmanovsky, "Reference and command governors for systems with constraints: A survey on theory and applications," *Automatica*, vol. 75, pp. 306–328, 2017.
[20] P. Derler, E. A. Lee, and A. S. Vincentelli, "Modeling cyber—physical systems," *Proc. IEEE*, vol. 100, no. 1, pp. 13–28, Jan. 2012.
[21] M. Rungger and P. Tabuada, "A notion of robustness for cyber-physical systems," *IEEE Trans. Autom. Control*, vol. 61, no. 8, pp. 2108–2123, Aug. 2016.
[22] S. K. Khaitan and J. D. McCalley, "Design techniques and applications of cyberphysical systems: A survey," *IEEE Syst. J.*, vol. 9, no. 2, pp. 350–365, Jun. 2015.
[23] R. Tian, N. Li, A. Girard, and I. Kolmanovsky, "Controller mode and reference governor for constraint and failure management in vehicle platoon systems," in *Proc. IEEE Conf. Control Technol. Appl.*, 2020, pp. 660–665.
[24] F. Tedesco, A. Casavola, and R. Russo, "Plug-and-play distributed supervision schemes for decoupled interconnected dynamical systems," in *Proc. IEEE Conf. Decis. Control*, 2019, pp. 3527–3532.
[25] F. Tedesco and A. Casavola, "Turn-based command governor strategies for interconnected dynamical systems with time-varying couplings," in *Proc. Amer. Control Conf.*, 2020, pp. 4564–4569.
[26] F. Tedesco, A. Casavola, and E. Garone, "Distributed command governor strategies for constrained coordination of multi-agent networked systems," in *Proc. Amer. Control Conf.*, 2012, pp. 6005–6010.
[27] A. Casavola, E. Garone, and F. Tedesco, "A parallel distributed supervision strategy for multi-agent networked systems," *Syst. Control Lett.*, vol. 97, pp. 115–124, 2018.
[28] A. Casavola, F. Tedesco, and E. Garone, "A distributed multi-agent command governor strategy for the coordination of networked interconnected systems," *IEEE Trans. Autom. Control*, vol. 59, no. 8, pp. 2099–2112, Aug. 2014.
[29] L. Barenboim and M. Elkin, *Distributed Graph Coloring: Fundamentals and Recent Developments*. Williston, VT, USA: Morgan and Claypool Publishers, 2013.
[30] R. Georgios, G. Gorman, and P. H. J. Kelly, "A fast and scalable graph coloring algorithm for multi-core and many-core architectures," in *Proc. Eur. Conf. Parallel Process.*, 2015, pp. 414–425.
[31] R. L. Brooks, "On colouring the nodes of a network," *Math. Proc. Cambridge Philos. Soc.*, vol. 37, pp. 194–197, 1941.
[32] E. G. Gilbert, I. Kolmanovsky, and K. T. Tan, "Discrete-time reference governors and the nonlinear control of systems with state and control constraints," *Int. J. Robot. Nonlinear Control*, vol. 5, no. 5, pp. 487–504, 1995.
[33] F. Tedesco and A. Casavola, "Distributed iterative command governor schemes for interconnected linear systems," *Int. J. Robust Nonlinear Control*, vol. 27, no. 18, pp. 4788–4807, 2014.
[34] E. Raboin, P. Švec, D. S. Nau, and S. K. Gupta, "Model-predictive asset guarding by team of autonomous surface vehicles in environment with civilian boats," *Auton. Robots*, vol. 38, pp. 261–282, 2015.
[35] S. Y. Luis, D. G. Reina, and S. L. T. Marín, "A deep reinforcement learning approach for the patrolling problem of water resources through autonomous surface vehicles: The Ypacarai lake case," *IEEE Access*, vol. 8, pp. 204076–204093, 2020.
[36] L. Dai and K. J. Åström, "Dynamic matrix control of a quadruple tank process," *IFAC Proc. Volumes*, vol. 32, pp. 6902–6907, 1999.
[37] K. H. Johansson and J. L. R. Nunes, "A multivariable laboratory process with an adjustable zero," in *Proc. Amer. Control Conf*, 1998, pp. 2045–2049.
[38] M. Mercangöz and F. J. Doyle III, "Distributed model predictive control of an experimental four-tank system," *J. Process Control*, vol. 17, pp. 297–308, 2007.

Francesco Tedesco (Senior Member, IEEE) received the master's degree in automation engineering and the Ph.D. degree in systems and computer engineering from the University of Calabria, Rende, Italy, in 2008 and 2012, respectively.

In 2010, he was a Visiting Scholar Research with ETH Zurich, Zurich, Switzerland, and in 2014, he covered the same position with the Imperial College of London, London, U.K. From 2012 to 2017, he was an Assistant Researcher with DIMES Department, University of Calabria, where he has been an Assistant Professor since 2018, and an Associate Professor since 2022. His research interests include supervision approaches over data network, model predictive control, automotive applications, and control of cyber-physical systems.

Dr. Tedesco was a co-recipient of the Best Paper Award at the IEEE-CoDIT 2019 Conference, Paris, France. He currently an Associate Editor for *Asian Journal of Control*, *IET Control Theory and Application*, and IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING. Since 2022, he has been a Member of the IFAC Technical committee TC 6.4. Fault Detection, Supervision and Safety of Techn. Processes-SAFEPROCESS. Within the same committee, he is a Member of the working group "Safety and Security of Cyberphysical Systems."

Alessandro Casavola received the Ph.D. degree in systems engineering from the University of Bologna, Bologna, Italy, in 1990.

From 1996 to 1998, he was a Researcher with the Department of Systems and Computer Engineering, University of Florence, Florence, Italy. Since 1998, he has been with the Department of Computer Science, Modeling, Electronics and Systems Engineering, University of Calabria, Rende, Italy, as an Associate Professor first and as Full Professor since 2005. His research interests include constrained predictive control, control under constraints, control reconfiguration for fault tolerant systems, and supervision approaches over data networks.

Dr. Casavola was the Program Chair of IEEE ISIC 2014 Symposium [within the IEEE Multi Conference on System and Control (IEEE MSC 2014)], held in NICE, France on 2014 and the Program Co-Chair of SysTol 2019, held in Casablanca, Morocco, Sep. 18–20, 2019. Since 2009, he has been a Subject Editor for the *International Journal of Adaptive Control and Signal Processing*, and from 2018 to 2022, he was an Associate Editor for the IEEE CONTROL SYSTEMS LETTERS. He was the co-recipient of the Journal of Process Control Best Survey Paper Award 2023 and the co-recipient of the 2023 IEEE Vehicle Power and Propulsion Conference (VPPC 2023) Best Paper Award.