Average Consensus over Directed Networks in Open Multi-Agent Systems with Acknowledgement Feedback

Evagoras Makridis[1], Andreas Grammenos[2], Gabriele Oliva[3], Evangelia Kalyvianaki[2], Christoforos N. Hadjicostis[1], and Themistoklis Charalambous[1]

Abstract— **In this paper, we address the distributed average consensus problem over directed networks in open multi-agent systems (OMAS), where the stability of the network is disrupted by frequent agent arrivals and departures, leading to a time-varying average consensus target. To tackle this challenge, we introduce a novel ratio consensus algorithm (OPENRC) based on acknowledgement feedback, designed to be robust to agent arrivals and departures, as well as to unbalanced directed network topologies. We demonstrate that when all active agents execute the OPENRC algorithm, the sum of their state variables remains constant during quiescent epochs when the network remains unchanged. By assuming eventual convergence during such quiescent periods following persistent variations in system composition and size, we prove the convergence of the OPENRC algorithm using column-stochasticity and sum-preservation properties. Finally, we apply and evaluate our proposed algorithm in a simulated environment, where agents are departing from and arriving in the network to highlight its resilience against changes in the network size and topology.**

Index Terms— **distributed average consensus, open multi-agent systems, acknowledgement feedback.**

I. INTRODUCTION

In the domain of distributed and *multi-agent systems* (MAS), a plethora of research has been conducted on collaborative decision-making and consensus. MAS consists of interconnected computing nodes, or agents, that exchange information with the goal of collaboratively attaining a shared understanding or decision on network-wide parameters, despite each relying solely on local information and data from neighboring agents. The lack of global knowledge about the states of other agents or network-wide parameters highlights the distributed nature of these systems, which are particularly suitable for tasks beyond the capabilities of individual master entities or those more efficiently managed by a cluster of simpler agents.

Notably, average consensus is a fundamental problem with broad applications across various fields. The goal of average consensus algorithms is to ensure that individual agent states or decisions converge to a common value, specifically the average of their initial states (the sum of the agents' initial states is often referred to as total mass). While significant advancements have been made in traditional MAS environments with a static, predefined number of agents, the introduction of *Open Multi-Agent Systems* (OMAS) has added a new layer of complexity (refer to the recent thesis in [1] for an overview on the representations, limitations, and applications of OMAS). In OMAS settings, agents are allowed to dynamically join or leave the network during run-time, presenting challenges for maintaining stable consensus and effective collaboration.

Research efforts have spanned various aspects of OMAS dynamics, including trust and reputation models [2], gossiping frameworks [3], and the formation of short-term teams [4], [5]. Further exploration into algorithms that handle predetermined periods of agent arrivals and departures [6], and strategies for managing time-varying network sizes [7], alongside methods for estimating time-varying averages [8], [9], contribute to a rich body of knowledge aimed at enhancing the efficiency and reliability of MAS in a variety of applications. Additionally, the application to distributed optimization [10], [11] and learning [12] showcase the depth of research in addressing the complexities introduced by agent dynamics. Finally, it is worth mentioning the work in [13], where the problem of OMAS interaction among agents with nonlinear coupling is addressed.

While prior studies that have explored various consensus frameworks within OMAS heavily rely on the assumption that the underlying graph is undirected, in this work we aim at designing an average consensus algorithm that can handle agent arrivals and departures over directed networks of time-varying size. In particular, we consider OMAS that consist of a set of agents that changes over time, within which agents can exchange information over a *strongly connected*[1] and possibly unbalanced directed network. Within this setup, all active agents aim at iteratively computing the average of their joining masses with which they entered the network. Each agent updates its state with local computation, which is often expressed as a linear combination of the state variables received by its immediate in-neighbors[2].

Aiming towards the development of scalable average consensus algorithms that adapt to dynamic agent participation,

[1]Department of Electrical and Computer Engineering, School of Engineering, University of Cyprus, 2109 Aglantzia, Nicosia, Cyprus. E-mails: {makridis.evagoras, hadjicostis.christoforos, charalambous.themistoklis}@ucy.ac.cy.

[2]Department of Computer Science and Technology, University of Cambridge, Cambridge CB2 1TN, United Kingdom. E-mails: {ag926, ek264}@cl.cam.ac.uk.

[3]Department of Engineering, University Campus Bio-Medico of Rome, Via Alvaro del Portillo, 21–00128 Roma, Italy. E-mail: g.oliva@unicampus.it.

The work of E. Makridis and T. Charalambous was partly funded by the project MINERVA, which received funding from the European Research Council (ERC) under the European Union's Horizon 2022 research and innovation programme (Grant Agreement No. 101044629).

[1]A digraph is called strongly connected, if there exists a directed path between every pair of nodes in the graph.

[2]The in-neighbors of a node are all the nodes in the network from which this node can directly receive information.

our paper makes the following contributions.

- While previous works have primarily focused on undirected graphs, we introduce a fully distributed algorithm that can be executed by actively participating agents in OMAS over directed and possibly unbalanced graphs with one-bit acknowledgement feedback[3]. Our method eliminates the necessity of bidirectional information exchange to track a time-varying average consensus value, thus broadening its use in various applications.
- Unlike classical approaches in undirected graphs, the proposed algorithm does not require individual agents to maintain the cumulative mass received from their in-neighbors. Instead, agents exploit the values they received from their in-neighbors, when they last joined the network, to ensure that the total mass of the active agents in the network remains invariant in between stable network conditions. Such an approach, not only reduces the computational burden on individual agents and enhances the scalability in large-scale systems, but also ensures that the active agents in the network can eventually reach the average consensus value in between stable network conditions.

II. Preliminaries and Problem Setting

A. Open Network Model

In this work, we assume that agents are able to join and leave the network at will, provided that the network remains strongly connected at each time instant. In particular, we assume a finite set $\mathcal{V} = \{v_1, \cdots, v_n\}$ that captures all n agents potentially participating to the network. However, at each time step k, only a subset of the agents, denoted by $\mathcal{V}(k) \subseteq \mathcal{V}$, is *active*, and we assume that interactions occur solely among active agents. For analysis purposes, we denote the activation of the agents by a time-varying indicator vector $\boldsymbol{\alpha}(k) \in \{0,1\}^n$, where $\alpha_j(k) = 1$ if agent v_j is active at time step k, *i.e.,* , if $v_j \in \mathcal{V}(k)$, while $\alpha_j(k) = 0$, otherwise. Based on this activation vector, the interconnection topology of the active agents in the network is modeled by a (possibly unbalanced) time-varying digraph $\mathcal{G}(k) = \{\mathcal{V}(k), \mathcal{E}(k)\}$. The number of agents active in the network at time k is denoted by $n(k) = |\mathcal{V}(k)|$. The interactions between active agents are captured by the set of digraph edges $\mathcal{E}(k) \subseteq \mathcal{V}(k) \times \mathcal{V}(k)$. A directed edge denoted by $\varepsilon_{ji} \in \mathcal{E}(k)$ indicates that node v_j receives information from node v_i, *i.e.,* $v_i \rightarrow v_j$, at time step k. The set of all potential interactions is denoted by $\mathcal{E} = \cup_{k=0}^{\infty} \mathcal{E}(k)$. The nodes from which node v_j receives information at time step k are called in-neighbors of node v_j at time step k, and belong to the set $\mathcal{N}_j^{\text{in}}(k) = \{v_i \in \mathcal{V}(k) | \varepsilon_{ji} \in \mathcal{E}(k)\}$. The number of nodes in this in-neighborhood set is called the in-degree of node v_j and is denoted by $d_j^{\text{in}}(k) = |\mathcal{N}_j^{\text{in}}(k)|$. The nodes that receive information from node v_j directly at time step k are called out-neighbors of node v_j at time step k, and belong to the set $\mathcal{N}_j^{\text{out}}(k) = \{v_l \in \mathcal{V}(k) | \varepsilon_{lj} \in \mathcal{E}(k)\}$. The number of nodes in its out-neighborhood set of node v_j at time step k is called the out-degree at time step k and is denoted by $d_j^{\text{out}}(k) = |\mathcal{N}_j^{\text{out}}(k)|$. The potential out-neighbors of node v_j is given by $\mathcal{N}_j^{\text{out}} = \cup_{k=0}^{\infty} \mathcal{N}_j^{\text{out}}(k)$. Clearly, each node $v_j \in \mathcal{V}(k)$ has immediate access to its own local state.

[3]Each acknowledgement feedback signal is sent over a narrowband error- and delay-free feedback channel. Hence, the network topology could still be assumed directed, although the feedback travels through undirected links.

The structure of the network at each time instant k can be formally defined by the following three sets that distinguish the operating modes of the agents.

Remaining: Agents that are currently active in the network at time k and are not departing in the next step $k+1$, belong to the set of remaining agents

$$\mathcal{R}(k) = \mathcal{V}(k) \cap \mathcal{V}(k+1).$$

Arriving: This set encompasses those agents that become active at time k and is denoted by $\mathcal{J}(k)$. In particular, we have that $\mathcal{J}(k)$ is given by the agents that are active at time $k+1$, but are inactive at time k, i.e,

$$\mathcal{J}(k) = \mathcal{V}(k+1) \setminus \mathcal{V}(k).$$

Departing: This set collects those agents that leave the network at time k. The set is denoted by $\mathcal{D}(k)$ and we have that

$$\mathcal{D}(k) = \mathcal{V}(k) \setminus \mathcal{V}(k+1).$$

Fig. 1 depicts shapshots of the network considered in an OMAS framework, at different time instants.

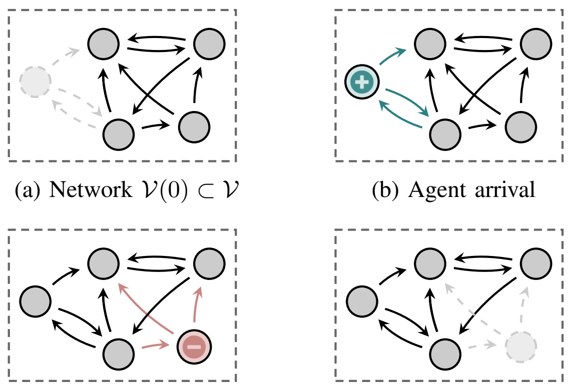

(a) Network $\mathcal{V}(0) \subset \mathcal{V}$ (b) Agent arrival
(c) Agent departure (d) Stable network

Fig. 1: OMAS with directional information flow. The size of the network changes as time progresses with agents joining the network, and present ones departing from the network.

B. Assumptions

In order to present the proposed problem statement, let us now state some assumptions.

Assumption 1 (Graph strong connectivity)**.** *The directed graph* $\mathcal{G}(k) = \big\{\mathcal{V}(k), \mathcal{E}(k)\big\}$ *is strongly-connected at all time instants* $k \geq 0$.

Assumption 2 (Acknowledgement feedback)**.** *For all times* k *and each link* $\varepsilon_{ji} \in \mathcal{E}(k)$*, we assume that agent* v_j *is able to transmit a one-bit acknowledgement message to* v_i.

Assumption 1 is common practice in the OMAS literature (connected for undirected graphs) [14], and, as discussed later in the paper, guarantees that the sum of the states of the active agents equals the sum of their initial states. As we will discuss later on, from a time instant k' when the network composition stops evolving (no agent departures/arrivals occur) one can relax the strong connectivity assumption on the digraph $\mathcal{G}(k)$ at time steps $k \geq k'$, by only requiring a sequence of jointly strongly connected graphs (refer to [15, Remark 3.4] for more information).

Assumption 2 is motivated by acknowledgement messages used by various protocols such as TCP, ARQ/HARQ, and ALOHA. Acknowledgement messages are utilized to confirm receipt of information and combat channel errors ensuring reliable transmissions over unreliable channels. These messages are assumed to be narrowband signals communicated over feedback channels, thus, their presence does not necessarily conflict with the directional data channel. In the context of average consensus, acknowledgement messages are often used to ensure convergence to the exact average consensus value in error-prone networks, and allow the agents to determine their out-degree at each time instant [16].

C. Problem Statement

Consider an open network as modeled in §II-A. The goal of the active agents $v_j \in \mathcal{V}(k)$ at time k is to collaboratively compute the time-varying average of their *joining masses*, *i.e.,* $\widehat{x}_j(k)$, albeit the changes in the composition and size of the network $\mathcal{V}(k)$. In particular, the target time-varying average consensus value of the active agents is given by

$$\bar{x}(k) = \frac{1}{|\mathcal{V}(k)|} \sum_{v_j \in \mathcal{V}(k)} \widehat{x}_j(k), \tag{1}$$

where $\widehat{x}_j(k)$ tracks the joining mass, *i.e.,* the mass of agent v_j when it last joins the network, defined as:

$$\widehat{x}_j(k) = \begin{cases} m_j(\max\{r \in \mathcal{K}_j \mid r \leq k\}), & \text{if } \alpha_j(k) = 1, \\ 0, & \text{otherwise,} \end{cases} \tag{2}$$

where $\mathcal{K}_j$ is the set of times when agent v_j (re)-joins the network, and $m_j(r) \in \mathbb{R}$ denote the *joining mass*, with which agent v_j enters the network at time $r \in \mathcal{K}_j$ (clearly, for $r \in \mathcal{K}_j$, we have $\alpha_j(r) = 1$ and $\alpha_j(r-1) = 0$). In other words, $\widehat{x}_j(k)$ is updated only when agent v_j becomes active and it remains constant and equal to the joining mass value with which it enters the network at the latest (largest) $r \in \mathcal{K}_j$ with $r \leq k$. When $v_j \notin \mathcal{V}(k)$, we can take $\widehat{x}_j(k) = 0$.

Clearly, asymptotic convergence in such a setting cannot be achieved since, with each departure or arrival of agents, a perturbation on the time-varying average consensus value is generated. Here, it is important to note that, in gossip-based algorithms, even when agents may approach some steady state behavior temporarily (between consecutive arrival/departure events), the average consensus value $\bar{x}(k)$ cannot be reached exactly. This is due to the incapability of such algorithms to eliminate instantaneously the outdated information about agents that depart from the network, as well as the absence of a global variable that tracks the current number of active agents $n(k)$ in the network.

In this work, however, we propose a fully distributed algorithm, hereinafter referred to as OPENRC, which is capable to instantaneously eliminate outdated information from agents that leave the network. Therefore, active agents drive their states towards the perturbed, yet correct, average consensus value (of the currently active agents), by utilizing the joining and departing mass of the agents departing the network. This information is broadcasted over directional data channels prior to an agent's departure, to its out-neighbors, with an appropriate weight that is assigned by exploiting the corresponding acknowledgement signals.

III. OPENRC ALGORITHM

To compute the time-varying average consensus value in (1), we follow the ratio consensus paradigm in which each agent v_j maintains two variables $x_j(k)$ and $y_j(k)$ (initialized upon activation at a given time r as $\widehat{x}_j(r)$ and to $\widehat{y}_j(r) = 1$, respectively). Here it is important to note that, $y_j(k)$ is an auxiliary variable which aims at compensating for the directional information flow (for more information, we refer the readers to [15]). First, suppose that agent v_j sends information to its out-neighbors $v_l \in \mathcal{N}_j^{\text{out}}(k)$ at iteration k. We assume that, at the beginning of each round k, the out-neighbors of v_j are able to send an acknowledgement feedback signal $f_{jl}(k)$ to v_j. In particular, for all the potential out-neighbors $v_l \in \mathcal{N}_j^{\text{out}}$, of node v_j, we have

$$f_{jl}(k) = \begin{cases} 1, & \text{if } v_l \in \mathcal{R}(k) \wedge v_j \in \mathcal{V}(k), \\ 0, & \text{if } v_l \in \mathcal{D}(k). \end{cases} \tag{3}$$

Based on the received feedback from each out-neighbor $v_l \in \mathcal{N}_j^{\text{out}}$, encoded by $f_{jl}(k)$, agent v_j is able to compute its currently active out-neighbors at each time k by

$$|\mathcal{N}_j^{\text{out}}(k) \cap \mathcal{R}(k)| = \sum_{v_l \in \mathcal{N}_j^{\text{out}}(k)} f_{jl}(k).$$

In turn, the above quantities allow each agent v_j to assign weights on its out-going links that scale the values to be sent to its currently active out-neighbors. Appropriate scaling of the out-going information is crucial to ensure that the total mass of the active agents in the network remains invariant in between stable network conditions. In what follows, we design the weight assignment from the perspective of node v_j, for its out-going links $\varepsilon_{lj} \in \mathcal{E}(k)$.

Arriving mode: When agent v_j arrives in the network, *i.e.,* $v_j \in \mathcal{J}(k)$, it simply registers its joining mass by setting

$$x_j(k+1) = \widehat{x}_j(k+1) \tag{4a}$$
$$y_j(k+1) = \widehat{y}_j(k+1) \tag{4b}$$
$$z_j(k+1) = x_j(k+1)/y_j(k+1), \tag{4c}$$

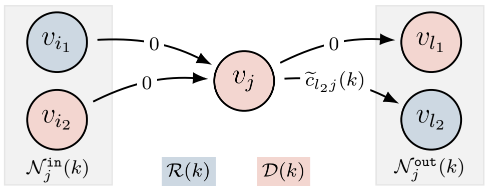
and starts interacting with active neighboring agents at the next time step.

Departing mode: When agent v_j departs from the network, *i.e.,* $v_j \in \mathcal{D}(k)$, it broadcasts $\widetilde{c}_{lj}(k)(x_j(k) - \widehat{x}_j(k))$ and $\widetilde{c}_{lj}(k)(y_j(k) - \widehat{y}_j(k))$ to the out-neighbors $v_l \in \mathcal{N}_j^{\text{out}}(k) \setminus$

$\mathcal{D}(k)$ that remain in the network, as shown in Fig. 2, with the weights $\widetilde{c}_{lj}(k)$ given by

$$\widetilde{c}_{lj}(k) = \frac{a_j(k)(1-a_j(k+1))f_{jl}(k)}{\sum_{v_l\in\mathcal{N}_j^{\text{out}}(k)} f_{jl}(k)} = \begin{cases} \frac{1}{|\mathcal{N}_j^{\text{out}}(k)\cap\mathcal{R}(k)|}, & \text{if } v_l\in\mathcal{N}_j^{\text{out}}(k)\setminus\mathcal{D}(k)\\ 0, & \text{otherwise.}\end{cases} \tag{5}$$

Note that $\tilde{c}_{lj} = 0$ if $v_j \notin \mathcal{D}(k)$. After a departing agent $v_j\in\mathcal{D}(k)$ broadcasts its values, it departs from the network and hence we can consider that $x_j(k+1)=0$, $y_j(k+1)=0$, and $z_j(k+1)=0$.



v_{i_1}
v_{i_2}
$\mathcal{N}_j^{\text{in}}(k)$
0
0
v_j
0
$\widetilde{c}_{l_2 j}(k)$
v_{l_1}
v_{l_2}
$\mathcal{N}_j^{\text{out}}(k)$
$\mathcal{R}(k)$
$\mathcal{D}(k)$

Fig. 2: Weight assignment of node $v_j\in\mathcal{D}(k)$. Nodes denoted in light blue belong in $\mathcal{R}(k)$, while the ones denoted in light red belong in $\mathcal{D}(k)$.

Remaining mode: When agent v_j remains in the network, *i.e.*, $v_j\in\mathcal{R}(k)$, it broadcasts $c_{lj}(k)x_j(k)$ and $c_{lj}(k)y_j(k)$ to its out-neighbors $v_l\in\mathcal{N}_j^{\text{out}}(k)\cap\mathcal{R}(k)$, as shown in Fig. 3, with the weights $c_{lj}(k)$ given by

$$c_{lj}(k) = \frac{a_j(k)a_j(k+1)f_{jl}(k)}{1+\sum_{v_l\in\mathcal{N}_j^{\text{out}}(k)} f_{jl}(k)} = \begin{cases} \frac{1}{1+|\mathcal{N}_j^{\text{out}}(k)\cap\mathcal{R}(k)|}, & \text{if } v_l\in\mathcal{M}_j(k),\\ 0, & \text{otherwise,}\end{cases} \tag{6}$$

where $\mathcal{M}_j(k) = \{\mathcal{N}_j^{\text{out}}(k)\cap\mathcal{R}(k)\}\cup\{v_j\}$. Note that $c_{lj}(k) = 0$ if $v_j\notin\mathcal{R}(k)$.

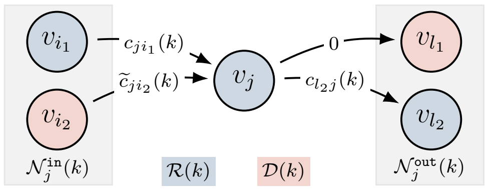


Fig. 3: Weight assignment of node $v_j\in\mathcal{R}(k)$. Nodes denoted in light blue belong in $\mathcal{R}(k)$, while the ones denoted in light red belong in $\mathcal{D}(k)$.

In general, at time step k, each active agent $v_j\in\mathcal{V}(k)$ broadcasts the values

$$\zeta_j^{(x)}(k) = c_{lj}(k)x_j(k)+\widetilde{c}_{lj}(k)(x_j(k)-\widehat{x}_j(k)), \tag{7}$$

$$\zeta_j^{(y)}(k) = c_{lj}(k)y_j(k)+\widetilde{c}_{lj}(k)(y_j(k)-\widehat{y}_j(k)), \tag{8}$$

to all its currently active out-neighbors. At the same time, upon the reception of information arrived within time slot k from the in-neighbors of v_j, each agent $v_j\in\mathcal{R}(k)$ updates its local variables as follows:

$$x_j(k+1) = \sum_{v_i\in\mathcal{I}_j^{\mathcal{R}}(k)} c_{ji}(k)x_i(k) + \sum_{v_i\in\mathcal{I}_j^{\mathcal{D}}(k)} \widetilde{c}_{ji}(k)\big(x_i(k)-\widehat{x}_i(k)\big), \tag{9a}$$

$$y_j(k+1) = \sum_{v_i\in\mathcal{I}_j^{\mathcal{R}}(k)} c_{ji}(k)y_i(k) + \sum_{v_i\in\mathcal{I}_j^{\mathcal{D}}(k)} \widetilde{c}_{ji}(k)\big(y_i(k)-\widehat{y}_i(k)\big), \tag{9b}$$

$$z_j(k+1) = x_j(k+1)/y_j(k+1). \tag{9c}$$

where $\mathcal{I}_j^{\mathcal{R}}(k) = \{\mathcal{N}_j^{\text{in}}(k)\cap\mathcal{R}(k)\}\cup\{v_j\}$ and $\mathcal{I}_j^{\mathcal{D}}(k) = \{\mathcal{N}_j^{\text{in}}(k)\cap\mathcal{D}(k)\}$.

Algorithm 1 provides a synoptic view of one step of the proposed procedure for all operational modes of node v_j.

Algorithm 1 – OPENRC iteration at agent v_j.

```
1:  Arriving mode: $v_j \in \mathcal{J}(k)$
2:      Compute: $x_j(k+1)$, $y_j(k+1)$, and $z_j(k+1)$ via (4)
3:  Departing mode: $v_j \in \mathcal{D}(k)$
4:      Receive feedback: $f_{jl}(k)$ from all $v_l \in \mathcal{N}_j^{\text{out}}(k)$
5:      Send feedback: $f_{ij}(k)$ to all $v_i \in \mathcal{N}_j^{\text{in}}(k)$
6:      Assign weights: as in (5).
7:      Broadcast: $\zeta_j^{(x)}(k)$ and $\zeta_j^{(y)}(k)$ to all $v_l \in \mathcal{N}_j^{\text{out}}(k)$
8:      Set: $x_j(k+1)=0$, $y_j(k+1)=0$, and $z_j(k+1)=0$
9:  Remaining mode: $v_j \in \mathcal{R}(k)$
10:     Receive feedback: $f_{jl}(k)$ from all $v_l \in \mathcal{N}_j^{\text{out}}(k)$
11:     Send feedback: $f_{ij}(k)$ to all $v_i \in \mathcal{N}_j^{\text{in}}(k)$
12:     Assign weights: as in (6).
13:     Broadcast: $\zeta_j^{(x)}(k)$ and $\zeta_j^{(y)}(k)$ to all $v_l \in \mathcal{N}_j^{\text{out}}(k)$
14:     Receive: $\zeta_i^{(x)}(k)$ and $\zeta_i^{(y)}(k)$ from all $v_i \in \mathcal{N}_j^{\text{in}}(k)$
15:     Compute: $x_j(k+1)$, $y_j(k+1)$, and $z_j(k+1)$ via (9)
```

Remark 1. *Observe that the acknowledgement feedback does not necessarily need to be sent regularly at every time step k. Instead, it is only needed when a change in the activation status of the agents is made, i.e., when $\boldsymbol{\alpha}(k+1)\neq\boldsymbol{\alpha}(k)$.*

IV. Eventual Correctness Analysis

Before discussing the convergence properties of the proposed algorithm, let us define the global variables that describe the evolution of the network variables. We start by collecting the local agents' variables x_j, y_j, $\widehat{x}_j$, and $\widehat{y}_j$, into the following vectors, respectively, as

$$\boldsymbol{x}(k) = \big(x_1(k),\ldots,x_n(k)\big)^\top,\ \widehat{\boldsymbol{x}}(k) = \big(\widehat{x}_1(k),\ldots,\widehat{x}_n(k)\big)^\top,$$
$$\boldsymbol{y}(k) = \big(y_1(k),\ldots,y_n(k)\big)^\top,\ \ \widehat{\boldsymbol{y}}(k) = \big(\widehat{y}_1(k),\ldots,\widehat{y}_n(k)\big)^\top.$$

Note that the values of inactive agents are also included in these variables, although (for the sake of anaysis only and without loss of generality) they are set to 0.

Now, we can rewrite the OPENRC algorithm in its vector-matrix form:

$$\boldsymbol{x}(k+1) = C(k)\boldsymbol{x}(k)+\widetilde{C}(k)\big(\boldsymbol{x}(k)-\widehat{\boldsymbol{x}}(k)\big)+W(k)\widehat{\boldsymbol{x}}(k+1)$$
$$\boldsymbol{y}(k+1) = C(k)\boldsymbol{y}(k)+\widetilde{C}(k)\big(\boldsymbol{y}(k)-\widehat{\boldsymbol{y}}(k)\big)+W(k)\widehat{\boldsymbol{y}}(k+1)$$

where $C(k)$ and $\widetilde{C}(k)$ are matrices formed by the collection of the weights c_{ji} and $\tilde{c}_{ji}$, and

$$W(k) = \mathtt{diag}(a_j(k+1)(1-a_j(k)))$$

is a diagonal matrix with diagonal entries $w_{jj}(k) = 1$ if $v_j \in \mathcal{J}(k)$, while $w_{jj}(k) = 0$, otherwise.

In the following theorem, we will show that the sum of the active agents' variables $x_j(k)$ and $y_j(k)$ are equal to the sum of their joining masses, which is preserved within the time intervals when the network is stable. With such a property at hand, we will show that active agents track the target average consensus value at all times $k \geq 0$.

Theorem 1. *Let Assumptions 1 and 2 hold true. Then, the following holds for all $k \geq 0$:*

$$\sum_{v_j \in \mathcal{V}(k)} x_j(k) = \sum_{v_j \in \mathcal{V}(k)} \widehat{x}_j(k), \tag{12a}$$

$$\sum_{v_j \in \mathcal{V}(k)} y_j(k) = \sum_{v_j \in \mathcal{V}(k)} \widehat{y}_j(k). \tag{12b}$$

Proof. We prove the result by induction and, since the proof is the same for both $x_j(k)$ and $y_j(k)$, we only consider the first variable, *i.e.,* $x_j(k)$. In order to apply our inductive argument, we observe that for $k = 0$, it trivially holds

$$\sum_{v_j \in \mathcal{V}(0)} x_j(0) = \sum_{v_j \in \mathcal{V}(0)} \widehat{x}_j(0). \tag{13}$$

Let us now assume that (12a) holds true for a given iteration k; we need to show that, it also holds true at $k+1$. In particular, noting that by removing the columns and rows corresponding to the inactive agents, we obtain a column-stochastic matrix $C(k)+\widetilde{C}(k)$, with which we have that

$$\begin{aligned}
&\sum_{v_j \in \mathcal{V}(k+1)} x_j(k+1) \\
&= \boldsymbol{\alpha}^\top(k+1)\boldsymbol{x}(k+1) \\
&= \boldsymbol{\alpha}^\top(k+1)\Big(C(k)\boldsymbol{x}(k)+\widetilde{C}(k)\big(\boldsymbol{x}(k)-\widehat{\boldsymbol{x}}(k)\big)+W(k)\widehat{\boldsymbol{x}}(k+1)\Big) \\
&= \sum_{v_j \in \mathcal{R}(k)} x_j(k) + \sum_{v_j \in \mathcal{D}(k)} \big(x_j(k)-\widehat{x}_j(k)\big) + \sum_{v_j \in \mathcal{J}(k)} \widehat{x}_j(k+1) \\
&= \sum_{v_j \in \mathcal{V}(k)} \widehat{x}_j(k) - \sum_{v_j \in \mathcal{D}(k)} \widehat{x}_j(k) + \sum_{v_j \in \mathcal{J}(k)} \widehat{x}_j(k+1) \\
&= \sum_{v_j \in \mathcal{R}(k)} \widehat{x}_j(k) + \sum_{v_j \in \mathcal{J}(k)} \widehat{x}_j(k+1) = \sum_{v_j \in \mathcal{V}(k+1)} \widehat{x}_j(k+1),
\end{aligned}$$

where we used the fact that

$$\mathcal{V}(k) = \mathcal{R}(k) \cup \mathcal{D}(k), \quad \mathcal{V}(k+1) = \mathcal{R}(k) \cup \mathcal{J}(k)$$

and that $\widehat{x}_j(k+1) = \widehat{x}_j(k)$ holds for all $v_j \in \mathcal{V}(k+1)$. This completes our inductive proof. □

Theorem 2. *Consider that there exists a time k' where the network stabilizes, i.e., no further departures and/or arrivals are allowed. Then the following holds for all $v_j \in \mathcal{V}(k')$ and all $k \geq k'$:*

$$\lim_{k \to \infty} z_j(k) = \lim_{k \to \infty} \frac{x_j(k)}{y_j(k)} = \frac{1}{|\mathcal{V}(k')|} \sum_{v_l \in \mathcal{V}(k')} \widehat{x}_l(k').$$

Proof. We start our proof by emphasizing that after time k', $\widetilde{C}(k)$ and $W(k)$ will be a zero for all $k \geq k'$. Hence, the OPENRC algorithm will reduce to the ratio consensus algorithm for closed systems with initial condition $\boldsymbol{x}(k')$ and $\boldsymbol{y}(k')$. In particular, by Theorem 1, we have that $\sum_{v_j \in \mathcal{V}(k')} x_j(k') = \sum_{v_j \in \mathcal{V}(k')} \widehat{x}_j(k')$ and $\sum_{v_j \in \mathcal{V}(k')} y_j(k') = \sum_{v_j \in \mathcal{V}(k')} \widehat{y}_j(k')$. Moreover, $C(k)$ is a column-stochastic matrix that is also primitive as long as the underlying digraph is strongly connected. Then, by utilizing the result in [17], we have that the summation of the individual iterations $\boldsymbol{x}$ and $\boldsymbol{y}$ is preserved and gives

$$\frac{\boldsymbol{\alpha}^\top(k)\boldsymbol{x}(k)}{\boldsymbol{\alpha}^\top(k)\boldsymbol{y}(k)} = \frac{\sum_{\ell \in \mathcal{V}(k')} x_\ell(k')}{\sum_{\ell \in \mathcal{V}(k')} y_\ell(k')} = \frac{\sum_{\ell \in \mathcal{V}(k')} \widehat{x}_\ell(k')}{|\mathcal{V}(k')|},$$

which implies that the ratio $z_j(k)$ converges asymptotically to the average consensus value of the agents in $\mathcal{V}(k')$. This completes our proof. □

V. Numerical Examples

In this section, we evaluate the performance of the proposed algorithm with numerical simulations. In what follows we illustrate the time-varying average consensus error, $e(k) := \| \operatorname{diag}\big(\boldsymbol{\alpha}(k)\big)\boldsymbol{z}(k) - \mathbf{1}\bar{x}(k)\|_2$, where $\boldsymbol{z}(k) = (z_1(k), \ldots, z_n(k))^\top$, over a network $\mathcal{G}(k)$ which is strongly connected at every $k \geq 0$. To gain some insights on the performance of the algorithm, we consider 100 initially active agents at time step $k = 0$, *i.e.,* $n(0) = |\mathcal{V}(0)| = 100$, from a total of $n = |\mathcal{V}| = 150$ potentially active agents. Each initially active agent $v_j \in \mathcal{V}(0)$ registers its joining mass $\widehat{x}_j(0) = m_j$ where m_j is assumed to be chosen uniformly at random in the interval $[1, 10]$, while its auxiliary variable is set to $\widehat{y}_j(0) = 1$. To illustratively highlight the time-varying nature of the average consensus value in OMAS, we assume that arriving agents $v_j \in \mathcal{J}(k)$ enter the network with a greater joining mass $\widehat{x}_j(k) = m_j$ where m_j is chosen uniformly at random in the interval $[10, 20]$.

In Fig. 4, we present the evolution with time of the ratio $z_j(k)$ of active agents $v_j \in \mathcal{V}(k)$. Between the time intervals $1 < k \leq 80$ and $101 < k \leq 180$, the network size may increase or decrease by 1 due to agent arrival or departure, with 10% in the former interval, and 20% in the latter. Between the time intervals $80 < k \leq 100$ and $180 < k \leq 200$, the network is assumed to be stable (no arrivals or departures). Note that, for ease of presentation, the ratio $z_j(k)$ of departing nodes is not shown in Fig. 4 after their departure. The target average consensus value of the currently active agents at time k, *i.e.,* $\bar{x}(k)$ is shown in light red color. As it can be seen from Fig. 4, the target average consensus value $\bar{x}(k)$ is tracked by the currently active agents, while at the time instants where agents arrive to or depart from the network, their ratio $z_j(k)$ is disturbed due to the change of the network mass.

In the upper plot of Fig. 5, the average consensus error decays in between stable network time intervals, while at the time instants where agents are arriving or departing, the average consensus error increases abruptly due to the change of the total mass of the network. However, we note that, during long enough time intervals (*i.e.,* $80 < k \leq 100$ and $180 < k \leq 200$) in which the network is stable, the average

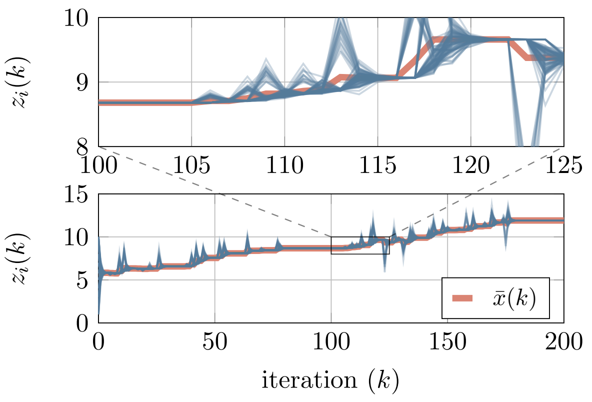


Fig. 4: Evolution with time k, of the active agents' ratios $z_i(k)$ using the OPENRC algorithm.

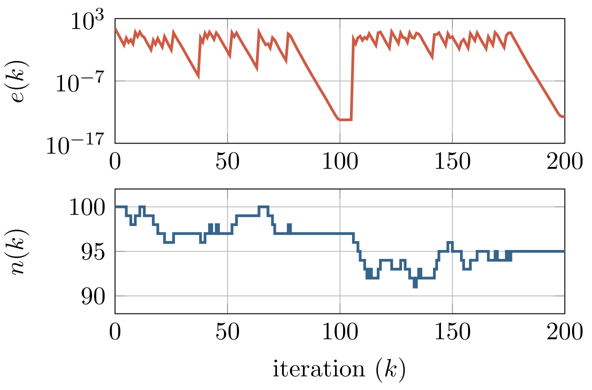


Fig. 5: The average consensus error, $e(k) = \| \operatorname{diag}\big(\boldsymbol{\alpha}(k)\big)\boldsymbol{z}(k) - \mathbf{1}\bar{x}(k)\|_2$, (upper); the number of active agents, $n(k)$, (lower), with time k.

consensus error reduces up to machine precision, until the next perturbation of the network composition.

VI. Conclusions

In this paper, we proposed a novel algorithm, called Open Ratio Consensus (OPENRC), to address the distributed average consensus problem within Open Multi-Agent Systems (OMAS) operating over directed and possibly unbalanced networks. Using acknowledgment feedback from out-neighbors, agents show resilience to frequent changes in the composition (*i.e.,* agent arrivals and departures) of the network. Our analysis establishes mass-preservation properties as long as the out-going weights assigned by each agent sum up to 1 at every time step. Such a property allows us to show that the active agents in the network converge to the exact average consensus value, given that after some time k' no further departures and/or arrivals occur. Our analysis is accompanied with numerical simulations that highlight the performance of the OPENRC algorithm in terms of minimizing the average consensus error within possibly short quiescent periods.

References

[1] C. M. d. G. de Carnières, "Open Multi-Agent Systems: Representation, Limitations and Decentralized Optimization," Ph.D. dissertation, Rice University, 2022.

[2] T. D. Huynh, N. R. Jennings, and N. R. Shadbolt, "An Integrated Trust and Peputation Model for Open Multi-Agent Systems," *Autonomous Agents and Multi-Agent Systems*, vol. 13, pp. 119–154, 2006.

[3] J. M. Hendrickx and S. Martin, "Open Multi-Agent Systems: Gossiping with Random Arrivals and Departures," in *IEEE Conference on Decision and Control*, 2017, pp. 763–768.

[4] F. Golpayegani, Z. Sahaf, I. Dusparic, and S. Clarke, "Participant Selection for Short-Term Collaboration in Open Multi-Agent Systems," *Simulation Modelling Practice and Theory*, vol. 83, pp. 149–161, 2018.

[5] F. Golpayegani, I. Dusparic, and S. Clarke, "Using Social Dependence to Enable Neighbourly Behaviour in Open Multi-Agent Systems," *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 3, pp. 1–31, 2019.

[6] J. M. Hendrickx and S. Martin, "Open Multi-Agent Systems: Gossiping with Deterministic Arrivals and Departures," in *54th Annual Allerton Conference on Communication, Control, and Computing*, 2016, pp. 1094–1101.

[7] C. M. de Galland, S. Martin, and J. M. Hendrickx, "Open Multi-Agent Systems with Variable Size: The Case of Gossiping," *arXiv preprint arXiv:2009.02970*, 2020.

[8] M. Franceschelli and P. Frasca, "Proportional Dynamic Consensus in Open Multi-Agent Systems," in *IEEE Conference on Decision and Control*, 2018, pp. 900–905.

[9] M. Franceschelli and P. Frasca, "Stability of Open Multi-Agent Systems and Applications to Dynamic Consensus," *IEEE Transactions on Automatic Control*, vol. 66, no. 5, pp. 2326–2331, 2021.

[10] J. M. Hendrickx and M. G. Rabbat, "Stability of Decentralized Gradient Descent in Open Multi-Agent Systems," in *IEEE Conference on Decision and Control*, 2020, pp. 4885–4890.

[11] N. Hayashi, "Distributed Subgradient Method in Open Multiagent Systems," *IEEE Transactions on Automatic Control*, pp. 6192–6199, 2022.

[12] T. Nakamura, N. Hayashi, and M. Inuiguchi, "Cooperative Learning for Adversarial Multi-Armed Bandit on Open Multi-Agent Systems," *IEEE Control Systems Letters*, pp. 1712–1717, 2023.

[13] G. Oliva, M. Franceschelli, A. Gasparri, and A. Scala, "A Sum-of-States Preservation Framework for Open Multi-Agent Systems With Nonlinear Heterogeneous Coupling," *IEEE Transactions on Automatic Control*, pp. 1991–1998, 2023.

[14] Z. A. S. Dashti, G. Oliva, C. Seatzu, A. Gasparri, and M. Franceschelli, "Distributed Mode Computation in Open Multi-Agent Systems," *IEEE Control Systems Letters*, vol. 6, pp. 3481–3486, 2022.

[15] C. N. Hadjicostis, A. D. Domínguez-García, and T. Charalambous, "Distributed Averaging and Balancing in Network Systems: with Applications to Coordination and Control," *Foundations and Trends® in Systems and Control*, vol. 5, no. 2-3, pp. 99–292, 2018.

[16] E. Makridis, T. Charalambous, and C. N. Hadjicostis, "Utilizing Feedback Channel Mechanisms for Reaching Average Consensus over Directed Network Topologies," in *IEEE American Control Conference*, 2023, pp. 1781–1787.

[17] A. D. Dominguez-Garcia and C. N. Hadjicostis, "Coordination and Control of Distributed Energy Resources for Provision of Ancillary Services," in *IEEE International Conference on Smart Grid Communications*, 2010, pp. 537–542.