

Finite time distributed averaging over gossip-constrained ring networks

Alessandro Falsone, Kostas Margellos, Simone Garatti, Maria Prandini

Abstract—We consider a multi-agent system where each agent has its own estimate of a given quantity and the goal is to reach consensus on the average. To this purpose, we propose a distributed consensus algorithm that guarantees convergence to the average in a finite number of communication rounds. The algorithm is tailored to ring networks subject to a gossip constraint. If the number of agents m is even, say $m = 2n$, then, the number of communication rounds needed is equal to n , which in this case is the diameter of the network, whereas it grows to $3n$ if the number of agents is odd and equal to $m = 2n + 1$.

Index Terms—Consensus, gossip algorithms, distributed averaging, networks.

I. INTRODUCTION

A typical problem encountered in a multi-agent system is that all agents are aiming at some common goal but they can communicate only with their neighbors to this purpose. Achieving a common goal often translates into reaching an agreement on the value taken by some quantity (the agreement variable) via some consensus algorithm. Basic consensus algorithms date back to [1]. Only in the last two decades, they have attracted the attention of both the computer science [2] and control engineering [3] communities.

There are different forms of consensus. Here, we are concerned with distributed averaging where each agent has its own estimate of a certain quantity and the goal is to reach consensus on the average of the values stored by all agents. The two most common approaches in the literature to the distributed averaging problem are linear iterations and gossip algorithms. The former is an iterative scheme where each agent updates its estimate of the average by taking a linear combination of its current estimate and those received by its neighboring agents, [4]. Gossip algorithms are similar, but they require that only pairwise communication occurs, [5], [6]. In both approaches, the interactions between agents can be modeled as a time-varying weighted graph, with vertices and edges respectively representing the agents and the communication links, and weights assigned to edges being the coefficients of the linear combinations. Under mild assumptions on the

coefficients and on the structure of the graph across iterations, both linear iterations schemes and gossip algorithms have been proven to asymptotically converge to the average, [4]. Since reaching consensus asymptotically can be limiting in practice, there is a fair amount of literature on how to analyze and optimize the convergence rate of both algorithms, see [4], [7] and references therein.

Alternatively, one can adopt an algorithm that reaches the exact average in finite time. The convergence rate can be assessed in terms of number of communication rounds, where a *communication round* is a time frame where possibly multiple communications occur simultaneously. This measure of convergence is practically quite meaningful since each communication round requires a certain amount of time, while internal computations are fast enough compared to communications to provide a negligible contribution.

In this paper we propose a finite time distributed consensus algorithm for gossip-constrained ring networks, i.e., networks that have a cycle graph with undirected edges and where only pairwise communications are allowed. We next review the main categories of finite time averaging algorithms in Section I-A, and then clarify the contribution of our work with respect to the existing literature in Section I-B.

A. Brief review of the literature on finite time averaging

Algorithms for finite time distributed averaging can be classified in three main categories: flooding-based algorithms where information is broadcasted across the network, routing-based algorithms where information is routed to one leading agent, and algorithms based on linear iterations, which can be further distinguished into linear iterations with fixed weights and linear iterations with time-varying weights. As pointed out in [2], [4], [6], which propose algorithms belonging to the first category, distributed average computation can be actually solved in a finite number of communication rounds by making each agent keep collecting all the values received and passing them to its neighbors. After a number of communication rounds equal to the diameter of the graph (i.e., the maximum distance between any two vertices), every agent knows all the values and can compute their average. This algorithm, albeit simple, presents two main issues: i) the number of values that each agent needs to store (and thus the memory usage) grows linearly with the number of agents, and ii) an unnecessary amount of information is exchanged which might overload the communication channels. These issues are even more critical when the quantities to be averaged are vectors instead of scalars. The memory requirement imposed by the approaches

Corresponding author: Alessandro Falsone.

Alessandro Falsone, Simone Garatti, and Maria Prandini are with Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, Milano, Italy (phone: +39 02 2399 {4028,3650,3441} e-mail: {alessandro.falsone, simone.garatti, maria.prandini}@polimi.it).

Kostas Margellos is with Department of Engineering Science, University of Oxford, Oxford, United Kingdom (phone: +44 1865 283912; e-mail: kostas.margellos@eng.ox.ac.uk).

This work is partially supported by the European Commission under the project UnCoVerCPS with grant number 643921.

of [2], [4], [6] can be alleviated at the expense of an increased number of communication rounds following [8], where the authors consider a fixed network topology and develop an algorithm based on a two-stage max-consensus scheme which achieves finite time convergence in $d(2m+1)$ communication rounds, where d is the diameter of the graph and m is the number of agents.

Other approaches that aim at limiting the memory requirements and the amount of information flowing in the network are those based on the availability of a routing mechanism. Typically, agents transmit all the information to a leader agent which performs the overall average and then broadcasts it back to all other agents, see e.g. convergecast, [9, Chapter 2]. In [10], [11], [12], the authors focus only on tree networks, i.e., graphs without loops, and show how to reach finite time convergence over these topologies. Note that, even though it is always possible to construct a spanning tree (i.e., a tree which reaches all vertices) from a connected graph by dropping some links, the diameter of the resulting tree might be greater than the diameter of the original graph, thus requiring more communication rounds to reach convergence.

Much effort has then been devoted in the literature to design simple algorithms (like linear iterations) that are able to solve the distributed averaging problem in finite time, see [13], [14], [15], [16], [17], [18], [19], [20] just to name a few. Most of these works take a global perspective, in that they assume to know the topology of the (undirected) graph, and then tune the weights so as to achieve finite time convergence.

The works in [13], [14], [15], [16], [17] focus on time-invariant networks with fixed weights. In [13] the solution to the finite time averaging problem is given in terms of the minimal polynomial of some matrix which gathers the weights and matches the graph topology. Convergence is achieved in $D+1$ communication rounds, where D is the degree of the minimal polynomial, provided that each agent stores all the $D+1$ previous values, which, however, might be impractical for large networks. In [14], the authors show that if the network is connected, then, for a class of fixed weights, the agents are able to reconstruct the values of the other agents and thus compute any function of the agents' initial values in a finite number of communication rounds. The work of [15] aims at finding the minimum number of communication rounds needed to construct the average by exploiting the trajectories of the local variable being updated according to a linear iteration scheme with non-designed weights. In [16] the authors focus on directed graphs and show that finite time average consensus can be achieved when the network is strongly connected and each agent knows the number of neighbors she is communicating with. All these approaches require the network to be time-invariant and the agents to store a certain amount of history of their local values. In [17] the authors prove that for specific type of graphs (distance-regular graphs) finite time averaging can be achieved in a number of communication rounds equal to the diameter of the graph.

In [18], [19] the network is also assumed to be time-invariant, but the weights (and thus the weight matrix) change across communication rounds. In [18] an analytic solution based on the joint diagonalization of the weight matrices is

provided, and it is shown that convergence can be achieved in a number of communication rounds equal to the number of distinct eigenvalues of the Laplacian matrix of the graph. In [19] the time-varying weights are designed based on the minimal polynomial of the adjacency matrix of the graph, and the proposed procedure is shown to converge in a number of communication rounds equal to the diameter of the graph for any distance-regular graph. In [20] the finite time averaging problem is treated as a nonlinear optimization program that can be solved numerically, where the weights in the graph are optimization variables. Finite time average can be achieved in this case with a number of communication rounds that ranges between d and $2d$, where d is the diameter of the network. However, agents have to solve a nonlinear program in a distributed fashion, which might be difficult from a computational point of view.

Finally, in [21], finite time averaging under the gossip constraint has also been investigated. In particular, it is shown that if the weights are constant and equal to $1/2$, then finite time convergence on undirected graphs can be achieved only if i) an agent can communicate with any other agent (i.e., any link can be activated if needed) and ii) the number of agents can be written as $m = 2^p$ for some p . In this case, finite time convergence can be achieved in $p = \log_2 m$ communication rounds, each one involving multiple disjoint pairwise communications.

B. Contribution of this paper

In this work we are concerned with the design of a finite time averaging algorithm specifically tailored to a ring topology, subject to a gossip constraint. Our approach is based on linear iterations with time-varying weights on a time-varying topology. By allowing the weights of the communication graph to be time-varying, we are able to prove finite time convergence for a network with an arbitrary number of agents. More specifically, for ring networks with an even number of agents, we propose a synchronous algorithm that converges in finite time equal to the diameter of the graph, which is also the lower bound on the number of communication rounds needed for a ring graph with no gossip constraint. As for ring networks with an odd number of agents, an algorithm based on the case of an even number of vertices is designed. Finite time convergence to the average is still guaranteed in a number of communication rounds that scales linearly with the number of agents. In this case, however, the lower bound is not attained. As for memory requirements, each agent needs to store only its estimate of the mean when the total number of agents is even, whereas the memory requirement is doubled in the odd case. In both cases, the required memory allocation is independent of the number of agents.

In comparison with algorithms based on flooding like [2], [4], [6] we impose fewer memory requirements that do not increase with the number of agents. In contrast to memory saving solutions like [8] and [9, Chapter 2] our approach achieves convergence in fewer communication rounds, without requiring any routing mechanism. As already discussed, tree-based strategies like [10], [11], [12] can be applied to connected graphs and account for gossip constraints (in particular,

see [10]). However, they present two main limitations: i) an additional distributed procedure is needed to construct the tree, and ii) the resulting graph has typically a larger diameter than the original connected one, resulting in a larger number of communication rounds required to achieve convergence. Due to the gossip constraint, works based on linear iterations with time-invariant topology and constant coefficients like [13], [14], [15], [16], [17] are not applicable. Contributions like [18], [19] achieving finite time convergence are based on a time-invariant topology but allow for time-varying coefficients. However, in both papers the coefficients are taken to be identical for all agents, and hence gossip communications are not allowed.

Finally, in contrast to the undirected case in [21], we allow for time-varying weights, each agent does not need to communicate with any other agent, and finite time convergence is attained on a ring network with an arbitrary number of agents, not only a power of two.

Note that our approach is tailored to the case of ring networks with synchronous communications, where the total number of agents m is known to everybody. This set-up can be achieved by some preliminary interactions between agents, see, e.g., [22] for some results in this direction.

C. Structure of the paper

The rest of the paper is organized as follows. In Section II, we introduce the problem set-up with the ring topology and the gossip constraint. The proposed distributed averaging algorithms for the cases of even and odd number of agents are described in Section III, where finite time convergence and related bounds on the number of communication rounds are also shown. Section IV presents two numerical examples, and finally, Section V draws some concluding remarks.

II. PROBLEM SETUP

We address the case of a multi-agent system characterized by a ring communication network, where each agent can communicate only with one of each neighbors at a time (gossip constraint). The agents are m in total and each one is identified by an integer i taking values in $\{1, 2, \dots, m\}$. Each agent i has its own estimate $x_i(0)$ of some quantity of interest. The goal is to devise a distributed algorithm through which each agent is able to compute the average

$$\bar{x} = \frac{1}{m} \sum_{i=1}^m x_i(0), \quad (1)$$

in a finite number of communication rounds only by exchanging information with its neighbors under the gossip communication protocol defined in the sequel. Evidently, only the case with $m > 2$ is of interest.

The network communication structure can be represented as an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \dots, m\}$ is the set of vertices representing the agents, and \mathcal{E} is the set of edges, defined as the following collection of ordered pairs of vertices:

$$\mathcal{E} = \bigcup_{i=1}^{m-1} \{(i, i+1), (i+1, i)\} \cup \{(1, m), (m, 1)\},$$

according to the ring topology. Since we are dealing with an undirected graph, with a slight abuse of notation we will use (i, j) to denote both edges (i, j) and (j, i) . According to this notation, for $m > 2$, the number of edges is even if m is even, and odd otherwise. The consecutive numbering of the vertices in the cycle graph is introduced to ease the algorithm description, e.g., by simplifying the definition of the predecessor and successor neighbor.

We next define the communication protocol used by the agents to exchange information over the network. We are here concerned with the design of a synchronous pairwise communication strategy. As introduced in Section I, the agents communicate in rounds. At each round, every agent is allowed to communicate only with one of its neighbors (gossip constraint). Since communication channels are represented by the edges of \mathcal{G} , the design of the communication protocol reduces to specifying a proper sequence of edges to activate, where edge (i, j) is said to be *active* at a given round if agents i and j exchange information at that round. As in the multigossip framework [11], more edges may be active at the same communication round, as long as they do not have any vertex in common so as to comply with the pairwise communication constraint.

To reduce the number of communication rounds we need to parallelize as much as possible the number of simultaneous pairwise communications. This can be interpreted as an edge-coloring problem on \mathcal{G} , [11], where edges with the same color represent communication channels which can be active at the same time. For a ring communication network with an even number $m > 2$ of vertices we need just two colors to minimize the number of communication rounds needed for all edges to be activated while satisfying the gossip constraint. In the odd case we need at least three colors. Figure 1 shows the coloring scheme for the even case in a pictorial form, with two groups of edges characterized by two different colors, blue and red, that can be activated alternatively in subsequent rounds, e.g., all the blue straight edges are activated at rounds $1, 3, \dots$, whereas the red wavy ones at rounds $2, 4, 6, \dots$.

It is worth noticing that in our approach, for a ring network with an even number of agents, the number of communication rounds coincides with the number of iterations of the proposed algorithm, as is usually the case in linear iteration schemes. Instead, for a ring network with an odd number of agents, this is no longer the case since each agent is artificially split into two subagents connected by a virtual link to get back to the even number of agents case. In this case, however, the iterations of the proposed algorithm that involve subagents communicating with their subagent neighbor actually encompass three communication rounds in order to meet the gossip constraint (see Section III-B).

Finally, we need to specify the weights associated with the edges, which define the linear update of the agents local estimate of the mean based on the information received from its neighbors. To this end, assume that at iteration k , agents i and j communicate. Agent i performs then the following update for its local estimate of the mean:

$$x_i(k) = (1 - \alpha_k)x_i(k-1) + \alpha_k x_j(k-1), \quad (2)$$

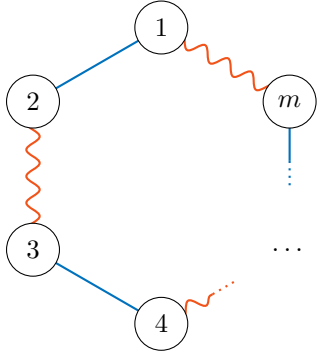


Fig. 1. Communication protocol under the gossip constraint in the case of an even number of agents: edges are grouped in two sets by a color coding. Only edges with the same color are active simultaneously.

where α_k is the (time-varying) weight associated with edge (i, j) at iteration k entering the convex update rule. Agent j performs an analogous update step.

The problem to be addressed is how to appropriately select α_k , $k = 1, 2, \dots$, so as to guarantee that there exists a finite iteration T such that $x_i(T) = \bar{x}$ for all $i = 1, \dots, m$, where \bar{x} is given by (1). Iteration k will possibly involve more than one communication rounds if needed to meet the gossip constraint.

III. PROPOSED SOLUTION

We next provide a solution to the finite time distributed averaging problem over a ring network topology with pairwise communication constraints as described in the previous section. In particular, we shall define which edges are activated at each communication round and appropriately set the values for the α_k weights in the update rule (2) to achieve finite time convergence of all agents' estimates to the average (1).

We will first address the case when the number of agents in the ring is even, and then we will extend the result to rings with an odd number of agents.

A. Ring with an even number of agents

For a ring with an even number of agents ($m = 2n$), the coefficients of the linear combination can be chosen according to the following rule:

$$\alpha_k = \begin{cases} \frac{k}{k+1}, & 1 \leq k < n \\ \frac{1}{2}, & k = n \\ 0, & k > n \end{cases} \quad (3)$$

which due to (2) entails that the estimate is kept constant after $k = n$ steps and hence the iterative process can be terminated.

The distributed overall procedure is obtained by making each agent run Algorithm 1 synchronously with the others. Algorithm 1 involves alternatively communicating with the predecessor and successor neighbor in the ring topology (according to the agent indexing shown in Figure 1), while updating its estimate according to (2) with the coefficients of the convex combination defined as in (3). Agents should agree on the neighbor to communicate with at the first iteration, i.e., which of the two edge groups of Figure 1 should be simultaneously activated. This choice is actually

Algorithm 1 Algorithm for agent $i - m$ even

```

1:  $x_i(0) \leftarrow$  initial value for agent  $i$ 
2: for  $k = 1$  to  $n$  do
3:   % Select a neighbor to communicate with
4:   if  $i + k$  is even then
5:      $j \leftarrow \text{Post}(i)$ 
6:   else
7:      $j \leftarrow \text{Pre}(i)$ 
8:   end if
9:   % Update the estimate using (2) and (3)
10:   $x_i(k) \leftarrow (1 - \alpha_k)x_i(k-1) + \alpha_k x_j(k-1)$ 
11: end for
12: return  $x_i(n)$ 

```

embedded in Algorithm 1 (line 4) and it is simply based on the agent identification number i . Functions $\text{Post}, \text{Pre}: \{1, 2, \dots, 2n\} \rightarrow \{1, 2, \dots, 2n\}$ in Algorithm 1 serve the purpose of specifying the successor and predecessor neighbor, respectively, and are given by:

$$\text{Post}(i) = \begin{cases} i + 1, & i = 1, \dots, 2n - 1 \\ 1, & i = 2n \end{cases} \quad (4)$$

and

$$\text{Pre}(i) = \begin{cases} 2n, & i = 1 \\ i - 1, & i = 2, \dots, 2n. \end{cases} \quad (5)$$

According to Algorithm 1, at iteration $k = 1$ agent $i = 1$ communicates with its successor agent 2, and, hence, the blue colored edges in Figure 1 are activated.

The following theorem holds for the proposed distributed scheme.

Theorem 1 (Finite time consensus). *Given a ring network with $m = 2n$ agents, suppose that all of them apply Algorithm 1 synchronously, with weights α_k , $k \geq 1$, defined as in (3). Then, after n communication rounds, the estimate x_i computed by each agent i , $i = 1, 2, \dots, 2n$, equals the average of their initial values (1).*

Proof. To ease the notation throughout the proof, we will use x_{i+s} as a shorthand for x_k , $k = \text{Post}(\text{Post}(\dots \text{Post}(i)))$ with $\text{Post}(\cdot)$ applied s times, and, similarly, x_{i-s} for $\text{Pre}(\cdot)$.

Our aim is to show that by applying Algorithm 1 to a ring with $2n$ agents we get

$$x_i(n) = \frac{1}{2n} \sum_{j=1}^{2n} x_j(0), \quad (6)$$

for every agent i , $i = 1, 2, \dots, 2n$.

Actually, it suffices to show that (6) holds for those i such that $i + n$ is even (i.e., i even if n is even, odd if n is odd). Indeed, according to Algorithm 1, agent $i + 1$ (for which $i + 1 + n$ is odd) communicates with agent i at step $k = n$ and its estimate satisfies

$$x_{i+1}(n) = (1 - \alpha_n)x_{i+1}(n-1) + \alpha_n x_i(n-1),$$

which is equal to $x_i(n) = (1 - \alpha_n)x_i(n-1) + \alpha_n x_{i+1}(n-1)$ given that $\alpha_n = \frac{1}{2}$. Hence, if $x_i(n)$ satisfies (6), then $x_{i+1}(n)$ does as well.

We shall focus then on agent i such that $i + n$ is even, and prove that the following equation hold

$$x_i(n) = \frac{1}{2n} \sum_{j=i-s+1}^{i+s} x_j(n-s-1) + \frac{n-s}{2n} [x_{i-s}(n-s-1) + x_{i+s+1}(n-s-1)], \quad (7)$$

for all $s = 1, \dots, n-1$. Note that, by substituting $s = n-1$ in (7), we get (6), which concludes the proof of the theorem.

The proof of equation (7) is by induction, i.e., we show that it is true for $s = 1$ (step 1), assume that it holds for some s (induction hypothesis), and then show that this is also the case for $s+1$ (step 2).

Step 1: By Algorithm 1,

$$x_i(n) = \frac{1}{2}x_i(n-1) + \frac{1}{2}x_{i+1}(n-1). \quad (8)$$

At step $k = n-1$, we have that $i+k = i+n-1$ is odd and $i+1+k = i+n$ is even, hence

$$x_i(n-1) = \frac{1}{n}x_i(n-2) + \frac{n-1}{n}x_{i-1}(n-2), \quad (9)$$

$$x_{i+1}(n-1) = \frac{1}{n}x_{i+1}(n-2) + \frac{n-1}{n}x_{i+2}(n-2), \quad (10)$$

and, by substituting (9) and (10) into (8), we obtain

$$x_i(n) = \frac{1}{2n} [x_i(n-2) + x_{i+1}(n-2)] + \frac{n-1}{2n} [x_{i-1}(n-2) + x_{i+2}(n-2)],$$

which is (7) with $s = 1$.

Step 2: For the sake of clarity we will treat the two terms in (7) separately: the summation first and then the other term.

First term: Let us start by considering the first two contributions in the summation, that is:

$$x_{i-s+1}(n-s-1) + x_{i-s+2}(n-s-1). \quad (11)$$

By Algorithm 1 when $k = n-s-1$ (and, hence, $i-s+1+k = i+n-2s$ is even and $i-s+2+k = i+n-2s+1$ is odd), we get

$$x_{i-s+1}(n-s-1) = \frac{1}{n-s}x_{i-s+1}(n-s-2) + \frac{n-s-1}{n-s}x_{i-s+2}(n-s-2), \quad (12)$$

$$x_{i-s+2}(n-s-1) = \frac{1}{n-s}x_{i-s+2}(n-s-2) + \frac{n-s-1}{n-s}x_{i-s+1}(n-s-2). \quad (13)$$

Substituting (12) and (13) into (11) we get that

$$x_{i-s+1}(n-s-1) + x_{i-s+2}(n-s-1) = x_{i-s+1}(n-s-2) + x_{i-s+2}(n-s-2). \quad (14)$$

The time-invariance property for the pairwise summation in (14) holds true for all other pairs in the summation of the first

term in (7), which always contains an even number of terms, i.e., $2s$. We can thus conclude that

$$\sum_{j=i-s+1}^{i+s} x_j(n-s-1) = \sum_{j=i-s+1}^{i+s} x_j(n-s-2). \quad (15)$$

Second term: Now consider the second term in (7), which is reported here for ease of reference:

$$\frac{n-s}{2n} [x_{i-s}(n-s-1) + x_{i+s+1}(n-s-1)]. \quad (16)$$

By Algorithm 1 when $k = n-s-1$ (and, hence, $i-s+k = i+n-2s-1$ is odd, while $i-s+1+k = i+n-2s$ is even), we have that

$$x_{i-s}(n-s-1) = \frac{1}{n-s}x_{i-s}(n-s-2) + \frac{n-s-1}{n-s}x_{i-s-1}(n-s-2), \quad (17)$$

$$x_{i+s+1}(n-s-1) = \frac{1}{n-s}x_{i+s+1}(n-s-2) + \frac{n-s-1}{n-s}x_{i+s+2}(n-s-2). \quad (18)$$

Substituting (17) and (18) into (16), (16) reduces to

$$\frac{1}{2n} [x_{i-s}(n-s-2) + x_{i+s+1}(n-s-2)] + \frac{n-s-1}{2n} [x_{i-s-1}(n-s-2) + x_{i+s+2}(n-s-2)]. \quad (19)$$

Finally, substituting (15) and (19) into (7), which holds true by the induction hypothesis, we have that

$$x_i(n) = \frac{1}{2n} \sum_{j=i-s}^{i+s+1} x_j(n-s-2) + \frac{n-s-1}{2n} [x_{i-s-1}(n-s-2) + x_{i+s+2}(n-s-2)],$$

which is the same expression as (7) with $s+1$ in place of s , thus concluding the proof by induction of (7). \square

Remark 1 (Speed of convergence). Based on [17], the minimum number of communication rounds needed to achieve finite time convergence on a fixed ring topology with $2n$ agents is equal to the graph diameter n . Under the gossip constraint, such a bound cannot be improved. This implies that the result of Theorem 1, which shows that we need exactly n communication rounds to converge to the average in ring networks with an even number of agents, is tight. To the best of our knowledge, this outperforms existing results in the literature.

Remark 2 (Efficiency). Since at each iteration every agent communicates with a single agent only, the information transmitted in the proposed distributed algorithm is limited compared to alternative solutions in the literature with either fixed or time-varying topology without gossip constraints.

Remark 3 (Interpretation). At iteration k , the update of agent i , $i = 1, \dots, m$, in (2) can be alternatively written as

$$x_i(k) = x_i(k-1) - \alpha_k(x_i(k-1) - x_j(k-1)). \quad (20)$$

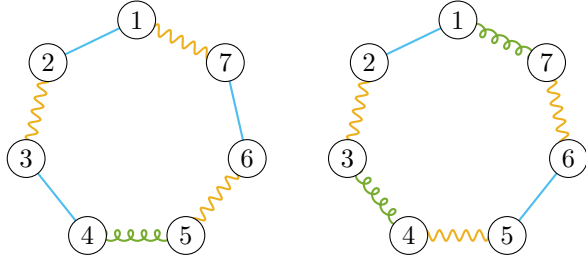


Fig. 2. Examples of communication protocols in a ring with an odd number of agents ($m = 7$): edges with the same colors are activated simultaneously. Three rounds are needed to activate all edges in both protocols.

By inspection of (20), it can be observed that the estimate of the average that each agent maintains evolves as a discrete time integrator, where the quantity that gets integrated is the mismatch between the communicating agents' estimates, weighted by α_k . This can be thought of as the evolution of a closed-loop dynamical system, with the feedback gain fixed according to (3). The interpretation of Theorem 1, is that the dynamical system (20), reaches \bar{x} at the n -th step, for all $i = 1, \dots, m$.

B. Ring with an odd number of agents

In the case of an odd number $m = 2n + 1$ of agents, we have to define a proper sequence of mode activations, thus choosing a coloring scheme for the ring. As stated earlier in Section II, for the odd case we need at least three colors for all edges to be activated in the minimum number of rounds while satisfying the gossip constraint. Whilst in the case of a ring with an even number of agents the coloring scheme is essentially unique (the solutions where two colors are swapped are indeed equivalent, see Figure 1), with three colors we can have multiple coloring configurations. To better clarify this observation, in Figure 2 we report two examples of a coloring scheme with three colors (cyan, yellow and green) for $m = 7$, which translates into two different multigossip sequences for the agents' communication. Interestingly, the solution that we propose in this paper does not depend on the adopted coloring scheme, which can be arbitrarily chosen but has to be agreed upon execution of the algorithm.

The solution for the case of an odd number of agents is based on the idea of making the number of agents even by considering each agent i as if it were composed by two subagents, say ia and ib , which can communicate with the predecessor and successor neighbor of agent i respectively, and are connected together by a *virtual communication link*. The communication link is virtual because it does not require

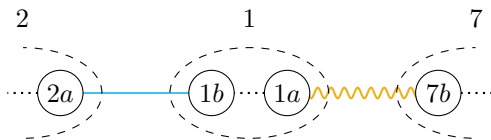


Fig. 3. Example of subagents and virtual communication links. Original agents are represented as dashed ellipsoids and virtual links as black dotted lines.

Algorithm 2 Algorithm for agent $i - m$ odd

```

1:  $x_{ia}(1) = x_{ia}(0) \leftarrow$  initial value for agent  $i$ 
2:  $x_{ib}(1) = x_{ib}(0) \leftarrow$  initial value for agent  $i$ 
3:  $k \leftarrow 2$ 
4: while  $k \leq m$  do
5:   % Communication rounds
6:   for  $r = 1$  to 3 do
7:     if edge  $(i, \text{Post}(i))$  is active then
8:       % Communicate with neighbor  $\text{Post}(i)$ 
9:        $j \leftarrow \text{Post}(i)$ 
10:       $x_{ib}(k) \leftarrow (1 - \alpha_k)x_{ib}(k-1) + \alpha_k x_{ja}(k-1)$ 
11:     else if edge  $(i, \text{Pre}(i))$  is active then
12:       % Communicate with neighbor  $\text{Pre}(i)$ 
13:        $j \leftarrow \text{Pre}(i)$ 
14:       $x_{ia}(k) \leftarrow (1 - \alpha_k)x_{ia}(k-1) + \alpha_k x_{jb}(k-1)$ 
15:     else
16:       % Stand by
17:     end if
18:   end for
19:    $k \leftarrow k + 1$ 
20:   % Update local estimates
21:    $x_{ia}(k) \leftarrow (1 - \alpha_k)x_{ia}(k-1) + \alpha_k x_{ib}(k-1)$ 
22:    $x_{ib}(k) \leftarrow (1 - \alpha_k)x_{ib}(k-1) + \alpha_k x_{ia}(k-1)$ 
23:    $k \leftarrow k + 1$ 
24: end while
25: return  $x_{ia}(m)$ 

```

any actual communication, but just a computation step for agent i . Figure 3 represents a pictorial view of this principle with reference to agent $i = 1$ of the ring network on the left of Figure 2. Original agents are represented as dashed ellipsoids and virtual links as black dotted lines.

Algorithm 2 describes the steps performed by agent i . Note that each agent maintains two estimates that are updated based on the information received from its successor and predecessor neighbor (steps 10 and 14) and are then combined when the virtual edge is activated (steps 21 and 22). Note also that α_k in Algorithm 2 must be chosen according to (3) with m in place of n .

Steps 10 and 14 require three communication rounds to be performed due to the gossip constraint (see the loop defined by steps 6-18 in Algorithm 2). In particular, in order for agent i to perform step 10 of Algorithm 2, she needs to wait the communication round in which the edge $(i, \text{Post}(i))$ is active. Same reasoning applies to step 14 of Algorithm 2, which requires to wait the communication round in which the edge $(i, \text{Pre}(i))$ is active. Finally, in one of the three rounds agent i is just stand by, whilst some other pairs of agents are communicating. Consider for example the communication protocol described in the left panel of Figure 2 with links activated in the following order: 1) cyan straight, 2) yellow wave, and 3) green spring. Then, in the first round ($r = 1$ in Algorithm 2) agent $i = 1$ communicates with agent $\text{Post}(1) = 2$, in the second round ($r = 2$) with agent $\text{Pre}(1) = 7$, and in the third round ($r = 3$) with no agent.

Theorem 2. *Given a ring network with $m = 2n + 1$ agents, suppose that all of them apply Algorithm 2 synchronously, with weights α_k , $k \geq 1$, defined as in (3) with n replaced by m . Then, after $3n$ communication rounds, the estimates x_{ia} and x_{ib} computed by each agent i , $i = 1, 2, \dots, m$, equal the average of their initial values (1).*

Proof. Consider the virtual network where each agent i is composed by two subagents, say ia and ib , which can communicate with the predecessor and successor neighbor of agent i respectively, and are connected by a virtual communication link. The topology of the virtual network is still a ring, but it now has an even number (i.e., $2m$) of agents. Moreover, by setting

$$x_{ia}(0) = x_{ib}(0) = x_i(0), \quad (21)$$

the average of the initial values of the $2m$ subagents equals the average of the initial values of the original m agents. By relabeling the subagents with numbers from 1 (for subagent 1a) to $2m$ (for subagent mb), one can recast this problem to the framework of Section III-A and apply Theorem 1 to complete the proof.

To this purpose, three observations are in order:

- 1) The initialization steps 1 and 2 are equivalent to steps 21 and 22 with $k = 1$ due to (21). In principle, these latter two steps would involve a communication between subagent ia and subagent ib , which, however, is virtual since subagent ia and subagent ib are both the same, namely, agent i . This virtual communication performed synchronously by all agents would correspond to activate one set of links (e.g., blue straight links with reference to Figure 1) in the virtual ring network.
- 2) Consider steps 10 and 14 inside the for loop of Algorithm 2. After three communication rounds (indexed by $r = 1, \dots, 3$), each subagent ib has communicated with subagent ja , with $j = \text{Post}(i)$. These 3 communication rounds occur within the same iteration k , since k is incremented only after the for loop, and correspond to activating the other set of links (e.g., red wavy links with reference to Figure 1) in the virtual ring network.
- 3) Consider now steps 21 and 22. As pointed out already in observation 1, no actual communication is needed between subagent ia and ib , and this virtual communication corresponds to activating the blue straight links in Figure 1 of the virtual ring network.

These three observations reveal that the execution of Algorithm 2 on the ring network with m agents is equivalent to the execution of Algorithm 1 on the virtual ring network with $2m$ agents, where the two networks have the same average of the initial values. By applying Theorem 1 to the virtual ring network, we obtain the average in a finite number m of iterations (indexed by k in Algorithm 2), alternating the edge activation as in Figure 1 with $2m$ in place of m , with the blue straight links being replaced by the black dotted lines representing the virtual links (no actual communication round), and the red wavy links representing the actual communication links, which with reference to Figure 2, are the cyan, the

yellow and the green ones, and have to be activated in three communication rounds to comply with the gossip constraint.

Among the m updates required by Algorithm 2 to converge, the first one corresponds to the initialization steps 1 and 2, and the remaining $m - 1 = 2n$ are performed inside the main loop of Algorithm 2. Since in each loop the index k is updated twice, we have that the code inside the main loop of Algorithm 2 is executed $(m-1)/2 = n$ times. Therefore we have n virtual communications (steps 21 and 22) and n actual communications involving 3 rounds each due to the gossip constraint (see steps 14 and 10). Thus, the overall number of communication rounds to achieve convergence to the average is $3n$, which concludes the proof. \square

IV. NUMERICAL EXAMPLES

In this section we apply our algorithms to two numerical examples: a ring network with an even number $m_e = 6$ of agents and a ring network with an odd number $m_o = 7$ of agents, so that $n = 3$ in both examples. In both cases, $x_i(0)$, $i = 1, \dots, m$ are scalar values.

In the even case, for all $i = 1, \dots, m_e$, we sampled $x_i(0)$ from a uniform distribution with support $[0, 1]$ and applied Algorithm 1 together with the communication protocol of Figure 1, with $m = m_e = 6$. Figure 4 shows the evolution of the $x_i(k)$ sequences across iterations. In accordance with Theorem 1, the average of the initial values (depicted as a red triangle) is reached in $n = 3$ iterations. Note that in this case iterations and communication rounds coincide.

In the odd case, for all $i = 1, \dots, m_o$, we sampled $x_i(0)$ from a uniform distribution with support $[0, 1]$, and applied Algorithm 2 together with the communication protocol depicted in Figure 2 (left panel). Figure 5 shows the evolution of the $x_{ia}(k)$ and $x_{ib}(k)$ sequences across iterations (sequences related to the same agents have the same color, and that of agent 1 is highlighted). In accordance with Theorem 2, the average of the initial values (depicted as a red triangle) is reached in $m_o = 7$ iterations and only $3n = 9$ communications rounds are required. Note that in this case iterations and communication rounds do not coincide: at iteration $k = 2$ of Algorithm 2 we have three communication rounds taking place. Indeed, not all values change at the same time; the ones corresponding to agents that are not communicating remain constant, whereas at iteration $k = 3$ the virtual links are active, implying that all values are updated locally and no real communication takes place.

V. CONCLUSIONS

In this paper we proposed an algorithm for finite time distributed averaging in the case of a ring network of agents, subject to a gossip constraint on communications. Interestingly, if the number of agents is even, consensus to the actual average is achieved in the minimum possible number of iterations, i.e., the diameter of the network, whereas the number of iterations needed in the case where the number of agents is odd is higher, but still finite. Numerical examples were also presented to demonstrate the efficacy of the proposed solution. Besides being of interest on its own, the proposed

algorithms can also be embedded in distributed optimization schemes where computing the average is instrumental to solving the optimization problem (see e.g. [23]) and finite time convergence is then a requirement. These optimization schemes have been further developed and tailored to large scale systems in new application areas like energy system [24], which could benefit from a distributed finite time averaging algorithm.

Admittedly, our algorithm is tailored to ring networks. The extension to more general network structures is not straightforward – except for the case where a cycle containing all vertices can be detected, which is however an NP-complete problem, [25, Chapter 9] – and requires additional investigation. Also, we assume synchronous communications as most of the literature on distributed averaging. Allowing for asynchronous communication protocols is a further interesting topic of research.

REFERENCES

- [1] J. N. Tsitsiklis, “Problems in decentralized decision making and computation,” Ph.D. dissertation, 1984.
- [2] N. A. Lynch, *Distributed algorithms*. Morgan Kaufmann, 1996.
- [3] A. Jadbabaie, J. Lin, and A. S. Morse, “Coordination of groups of mobile autonomous agents using nearest neighbor rules,” *IEEE Transactions on Automatic Control*, vol. 48, no. 6, pp. 988–1001, 2003.
- [4] L. Xiao and S. Boyd, “Fast linear iterations for distributed averaging,” *Systems & Control Letters*, vol. 53, no. 1, pp. 65–78, 2004.
- [5] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, “Gossip algorithms: Design, analysis and applications,” in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, vol. 3. IEEE, 2005, pp. 1653–1664.
- [6] B. J. Liu, S. Mou, A. S. Morse, B. Anderson, and C. Yu, “Deterministic gossiping,” *IEEE Proceedings*, vol. 99, no. 9, pp. 1505–1524, 2011.
- [7] A. Olshevsky and J. N. Tsitsiklis, “Convergence speed in distributed consensus and averaging,” *SIAM Journal on Control and Optimization*, vol. 48, no. 1, pp. 33–55, 2009.
- [8] G. Oliva, R. Setola, and C. N. Hadjicostis, “Distributed finite-time average-consensus with limited computational and storage capability,” *IEEE Transactions on Control of Network Systems*, no. 99, 2016.
- [9] H. Attiya and J. Welch, *Distributed computing: fundamentals, simulations, and advanced topics*. John Wiley & Sons, 2004, vol. 19.
- [10] C.-K. Ko and X. Gao, “On matrix factorization and finite-time average-consensus,” in *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*. IEEE, 2009, pp. 5798–5803.
- [11] S. Mou and A. Morse, “Finite-time distributed averaging,” in *American Control Conference (ACC), 2014*. IEEE, 2014, pp. 5260–5263.
- [12] J. M. Hendrickx, G. Shi, and K. H. Johansson, “Finite-time consensus using stochastic matrices with positive diagonals,” *IEEE Transactions on Automatic Control*, vol. 60, no. 4, pp. 1070–1073, 2015.
- [13] S. Sundaram and C. N. Hadjicostis, “Finite-time distributed consensus in graphs with time-invariant topologies,” in *American Control Conference, 2007. ACC’07*. IEEE, 2007, pp. 711–716.
- [14] —, “Distributed function calculation and consensus using linear iterative strategies,” *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 4, pp. 650–660, 2008.
- [15] Y. Yuan, G.-B. Stan, L. Shi, M. Barahona, and J. Goncalves, “Decentralised minimum-time consensus,” *Automatica*, vol. 49, no. 5, pp. 1227–1235, 2013.
- [16] T. Charalambous, Y. Yuan, T. Yang, W. Pan, C. N. Hadjicostis, and M. Johansson, “Distributed finite-time average consensus in digraphs in the presence of time delays,” *IEEE Transactions on Control of Network Systems*, vol. 2, no. 4, pp. 370–381, 2015.
- [17] J. M. Hendrickx, R. M. Jungers, A. Olshevsky, and G. Vankeerberghen, “Graph diameter, eigenvalues, and minimum-time consensus,” *Automatica*, vol. 50, no. 2, pp. 635–640, 2014.
- [18] A. Y. Kibangou, “Finite-time average consensus based protocol for distributed estimation over awgn channels,” in *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on*. IEEE, 2011, pp. 5595–5600.
- [19] —, “Step-size sequence design for finite-time average consensus in secure wireless sensor networks,” *Systems & Control Letters*, vol. 67, pp. 19–23, 2014.
- [20] L. Georgopoulos, “Definitive consensus for distributed data inference,” Ph.D. dissertation, École Polytechnique Fédérale de Lausanne, 2011.
- [21] G. Shi, B. Li, M. Johansson, and K. H. Johansson, “Finite-time convergent gossiping,” *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2782–2794, Oct 2016.
- [22] K. Yildirim, R. Carli, and L. Schenato, “Adaptive control-based clock synchronization in wireless sensor networks,” in *Proceedings of the European Control Conference ECC15*, 2015.
- [23] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and distributed computation: numerical methods*. Prentice-Hall, Inc., 1989.
- [24] A. D. Dominguez-Garcia and C. N. Hadjicostis, “Distributed algorithms for control of demand response and distributed energy resources,” in *2011 50th IEEE Conference on Decision and Control and European Control Conference*. IEEE, 2011, pp. 27–32.
- [25] R. E. Miller, J. W. Thatcher, and J. D. Bohlinger, *Complexity of Computer Computations*. Springer US, 1972.

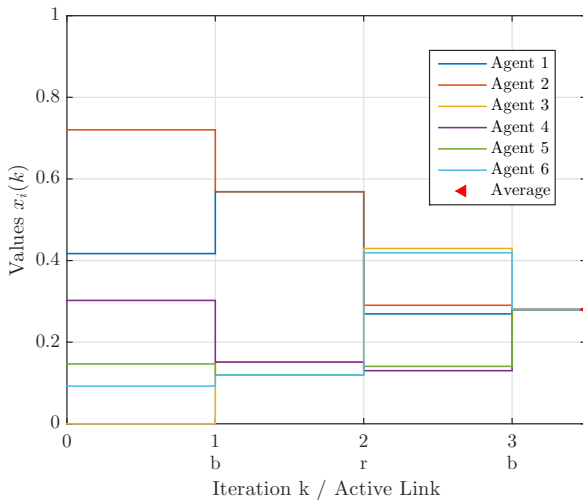


Fig. 4. Example of a ring network with $m_e = 6$ agents. On the horizontal axis we report both the iteration index k of Algorithm 1 and the active link (b: blue straight, r: red wavy) at the corresponding iteration.

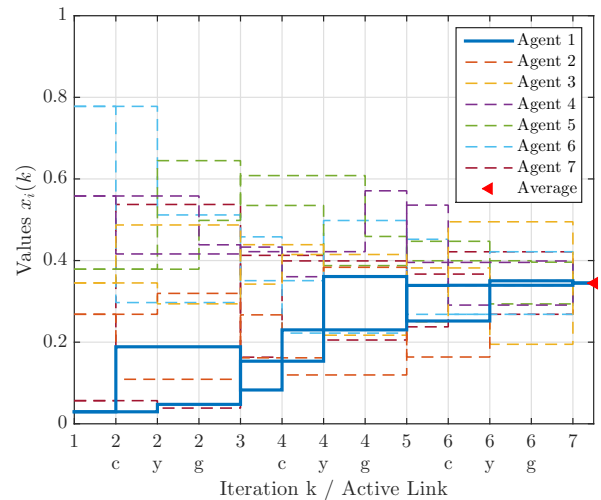


Fig. 5. Example of a ring network with $m_o = 7$ agents. On the horizontal axis we report both the iteration index k of Algorithm 2 and the active link (c: cyan straight, y: yellow wavy, g: green spring) at the corresponding iteration.