Consensus in Networked Multi-Agent Systems with Adversaries

Heath LeBlanc
heath.j.leblanc@vanderbilt.edu

Xenofon Koutsoukos
xenofon.koutsoukos@vanderbilt.edu

Institute for Software Integrated Systems
Department of Electrical Engineering and Computer Science
Vanderbilt University
Nashville, TN, USA

ABSTRACT

In the past decade, numerous consensus protocols for networked multi-agent systems have been proposed. Although some forms of robustness of these algorithms have been studied, reaching consensus securely in networked multi-agent systems, in spite of intrusions caused by malicious agents, or adversaries, has been largely underexplored. In this work, we consider a general model for adversaries in Euclidean space and introduce a consensus problem for networked multi-agent systems similar to the Byzantine consensus problem in distributed computing. We present the Adversarially Robust Consensus Protocol (ARC-P), which combines ideas from consensus algorithms that are resilient to Byzantine faults and from linear consensus protocols used for control and coordination of dynamic agents. We show that ARC-P solves the consensus problem in complete networks whenever there are more cooperative agents than adversaries. Finally, we illustrate the resilience of ARC-P to adversaries through simulations and compare ARC-P with a linear consensus protocol for networked multi-agent systems.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems; H.1.1 [**Models and Principles**]: Systems and Information Theory—*General Systems Theory*

General Terms

Algorithms, Security, Theory

Keywords

Consensus, Dynamic agent, Networked multi-agent system, Robustness, Adversary

1. INTRODUCTION

Reaching consensus is a fundamental problem in group coordination. The formal study of consensus has a rich history in management science [5] and distributed computing [18]. More recently, there has been a surge of research in the coordination of multi-agent networks. Within mobile robotics, there are several approaches researching the minimum attributes required to achieve distributed tasks such as gathering [1, 4, 29, 32]. In control, consensus algorithms have been used for the coordination of dynamic agents for group formation [10, 33], conflict resolution [24], and a host of other problems [19, 21]. In sensor networks, consensus has been considered for filtering [30], sensor fusion [36], and distributed hypothesis testing [22].

Various forms of uncertainty have been considered in consensus protocols for multi-agent networks. Reaching average consensus in a wireless network with interference is studied in [34]. Additive channel noise is addressed in [14]. Packet loss in ring networks is studied in [13]. Nonuniform time delays for a class of linear systems is considered in [17]. Contraction analysis is used in [35] to study nonlinear systems and wave variables are used in the communication for robustness to nonuniform constant delays. A virtual layer is used for self-stabilization of a network of robots to a desired curve in [11] whenever there are intermittent disturbances in the network. Robustness in terms of sensitivity to model uncertainty has been addressed in [15].

On the other hand, robustness of consensus protocols in networked multi-agent systems to malicious attacks and failures similar to the Byzantine failures of [16] has only been studied in the last few years. In [25–27], detecting and isolating malicious agents in discrete-time linear consensus networks is considered. Similarly, [31] addresses calculating functions of the initial states of cooperative agents in discrete-time linear consensus networks in the presence of malicious agents. Similar to these works, this paper considers a problem that addresses security of consensus networks; however, the problem introduced here is formulated in continuous-time. Moreover, the algorithm described here is less computationally complex when implemented in discrete-time.

Specifically, in this paper we consider robustness of networked multi-agent systems to adversaries. The multi-agent system is comprised of two classes: cooperative and adversarial agents. The cooperative agents have first-order integrator dynamics and the only assumption about the behavior of the adversarial agents is that their state trajectories are bounded and continuous. Additionally, we assume there is an upper bound on the number of adversarial agents present. The agents exchange scalar state information in an all-to-all manner either through communication or sensing. The goal is for the states of the cooperative agents to asymptotically align to a constant value within the range of their initial states, without knowledge of which agents are adversaries.

The main contribution of this work is the design and analysis of a consensus protocol that is robust to the presence of adversaries.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
HSCC'11, April 12–14, 2011, Chicago, Illinois, USA.
Copyright 2011 ACM 978-1-4503-0629-4/11/04 ...$10.00.

First, we introduce a system model in the context of ordinary differential equations (ODEs). Then, we define the adversarial agreement problem and present the consensus protocol, which we refer to as the Adversarially Robust Consensus Protocol (ARC-P). ARC-P is inspired by the *ConvergeApproxAgreement* algorithm [6, 18] and the linear consensus protocol (LCP) [23]. We prove that ARC-P yields a unique solution which solves the adversarial agreement problem with an exponential rate of convergence. Then, we show an upper bound on performance that allows for approximate solutions to the problem in finite time. Finally, we provide several simulation results comparing ARC-P to LCP in the presence of adversaries, and illustrate the performance tradeoff incurred by ARC-P for robustness to adversaries.

ARC-P combines ideas from the *ConvergeApproxAgreement* algorithm, which is resilient to Byzantine faults, and LCP. Specifically, it employs the sort and reduce function – which eliminate outlying values to ensure the output is within the range of cooperative agents' states – similarly to [6]. The concept of using a sum of relative state values (i.e., the difference between a neighboring agent's state and the given agent's state) as control input to a first-order integrator agent – in order to drive the agents' states together – is taken from LCP. By combining these ideas from distributed computing and control, we obtain a new consensus protocol that is resilient to adversaries, and can be analyzed using system theoretic techniques. Specifically, we analyze the Lipshcitz continuity of the protocol to ensure the uniqueness of solutions. We use error dynamics to show exponential pairwise convergence of the cooperative agent states. We show that the states of the cooperative agents always converge to a point within the range of their initial conditions using an invariant set argument. Finally, we bound the worst-case convergence time with respect to any arbitrarily small error tolerance, so the protocol can terminate in finite time with an approximate solution.

The paper is organized as follows: Section 2 describes the system model and the adversarial agreement problem. In Section 3 the protocol ARC-P is described. In Section 4, we prove that ARC-P solves the adversarial agreement problem. Section 5 presents simulations illustrating ARC-P in the presence of adversaries, and compares the performance of our solution with LCP. Related work is discussed in Section 6. Finally, Section 7 provides concluding remarks and some ideas for future work.

2. SYSTEM MODEL AND PROBLEM

2.1 System Model

The topology of the networked multi-agent system is described by a labelled strongly connected digraph, $\mathcal{D} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \dots, n\}$ describes the n dynamic agents. Without loss of generality, $\mathcal{V}$ is partitioned into a set of p cooperative agents, $\mathcal{V}_c = \{1, \dots, p\}$, and a set of q adversarial agents, $\mathcal{V}_a = \{p+1, \dots, n\}$, with $q = n - p$. The number of adversarial agents in the network is bounded by a constant $F \in \mathbb{Z}^+$, so that $q \leq F$. The edge set $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ models the information flow between the agents, which is realized either through communication or sensing. For each ordered pair $(i, j) \in \mathcal{E}$, state information flows from agent i to agent j. For loops, $(i, i) \in \mathcal{E}$ represents local state feedback. In this paper, the network is assumed to be complete, i.e., $\mathcal{E}{=}\{(i,j)|i,j \in \mathcal{V}\}$.

The networked multi-agent system is a composition of two interacting subsytems, i.e., the set of cooperative and adversarial agents. The agents interact in a synchronous manner by sharing state information, as shown in Figure 1. In the figure, $x_c = [x_1, \dots, x_p]^\mathsf{T} \in \mathbb{R}^p$ represents the states of the cooperative agents. Similarly, $x_a = [x_{p+1}, \dots, x_n]^\mathsf{T} \in \mathbb{R}^q$ represents the states of the adversaries. The state feedback to the adversarial agents is shown as dashed lines to indicate that this information may or may not be used to influence the behavior of the adversaries. On the other hand, the cooperative agents must use the state information from all the agents in a similar manner since the cooperative agents are unaware of which agents are adversaries. However, from a global perspective, the states of the adversaries can be viewed as uncertain inputs to the cooperative agents. This is the approach used to analyze the convergence properties of the subsystem of cooperative agents.

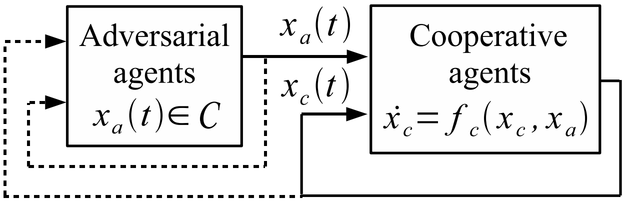


Figure 1: System model.

2.1.1 Cooperative Agents

Each cooperative agent $i \in \mathcal{V}_c$ has dynamics given by $\dot{x}_i = u_i$, where $u_i = f_i(x_c, x_a)$ is a control input, which is designed in such a way so that the cooperative agents reach consensus in spite of the influence of at most F adversaries. The state of the adversarial agents, x_a, is treated as an uncertain input; however, because there is no prior knowledge concerning which agents are adversarial, the control input must treat the state information from neighboring agents in the same manner. With these clarifications, the dynamics of the system of cooperative agents are given by

$$\dot{x}_c = f_c(x_c, x_a), \quad x_c(0) = x_{c_0},\ x_a(t) \in \mathcal{C}, \tag{1}$$

where $f_c(x_c, x_a){=}[f_1(x_c, x_a) \ \dots \ f_p(x_c, x_a)]^\mathsf{T}$, $x_{c_0} \in \mathbb{R}^p$ is the vector of initial values of the cooperative agents, and $\mathcal{C} \subset \mathbb{R}^q$ is some fixed compact set.

2.1.2 Adversarial Agents

The adversarial agents are assumed to be designed for the purpose of disrupting the objective of the cooperative agents. The main limitation on the behavior of the adversaries is that the state trajectory of each agent is restricted to bounded continuous functions of time. Specifically, we assume that $x_a(t)$ has a continuous trajectory that remains in some arbitrarily large, but fixed compact set $\mathcal{C} \subset \mathbb{R}^q$ for all $t \geq 0$. Although these assumptions eliminate most unstable systems, the fixed compact set $\mathcal{C}$ can be chosen large enough to include any finite region. One interesting case is when the adversaries are designed to drive the states of the cooperative agents to some unsafe region.

2.2 Adversarial Agreement Problem

Consider a networked multi-agent system consisting of n agents, where a subset of the agents are adversaries. Assume there exists an upper bound F on the number of such agents. Then the *adversarial agreement problem* is defined by two conditions: agreement and validity.

The *agreement condition* states that the pairwise absolute difference between the states of the cooperative agents approaches zero asymptotically, regardless of the adversaries' trajectories. That is, for all $x_c(0) \in \mathbb{R}^p$,

$$\lim_{t \to \infty} |x_i(t) - x_j(t)| = 0, \quad \forall i, j \in \mathcal{V}_c,\ x_a(t) \in \mathcal{C}. \tag{2}$$

Equivalently, the cooperative agents achieve the agreement condition if the state of the cooperative agents, x_c, converges to the agreement space, $\mathcal{A} = \{y \in \mathbb{R}^p | y_i = y_j, \forall i, j \in \mathcal{V}_c\}$.

The *validity condition* states that the limit of the state trajectory of each cooperative agent exists and is contained in the interval formed by the initial states of cooperative agents, regardless of the adversaries' trajectories. That is, if we define the interval

$$\mathcal{I}_0 = [\min_{i \in \mathcal{V}_c} x_i(0), \max_{j \in \mathcal{V}_c} x_j(0)],$$

then the validity condition is formulated as

$$\lim_{t \to \infty} x_i(t) \in \mathcal{I}_0, \quad \forall i \in \mathcal{V}_c,\ x_a(t) \in \mathcal{C}. \tag{3}$$

As in the case of the agreement condition, the validity condition can be stated in terms of x_c. Let $\mathcal{H}_0 = \mathcal{I}_0^p \subset \mathbb{R}^p$ denote the hypercube formed by the Cartesian product of p copies of $\mathcal{I}_0$. Then the validity condition stated in (3) is equivalent to $lim_{t \to \infty} x_c(t) \in \mathcal{H}_0$ for all $x_a(t) \in \mathcal{C}$. Note that if the system satisfies both the agreement and validity conditions, then all of the states of the cooperative agents will converge to a single limit point within $\mathcal{I}_0$.

Example: Consider the linear consensus protocol [23], which we will denote by LCP throughout this paper:

$$\dot{x}_i(t) = \sum_{j \in \mathcal{N}_i} (x_j(t) - x_i(t)),\ x_i(0) = x_{0_i}, \tag{4}$$

where $\mathcal{N}_i = \{j \in \mathcal{V} | (j, i) \in \mathcal{E}, j \neq i\}$. In complete networks, adversaries of the type outlined above cannot prevent the cooperative agents from asymptotically aligning their states to a consensus state. This is because for complete networks, (4) can be written as

$$\dot{x}_i(t) = -n x_i(t) + \sum_{j=1}^{n} x_j(t).$$

Therefore, the pairwise error $e_{ij} = x_i - x_j$ for $i, j \in \mathcal{V}_c$ has dynamics given by

$$\dot{e}_{ij}(t) = -n e_{ij}(t),$$

which converges exponentially to zero with rate n.

However, the validity condition is not satisfied by LCP. Even a single adversary in a complete network can become the leader and drive the state of each cooperative agent to an arbitrary point in the interval $\mathcal{C}$. Although LCP is designed for cooperative agents, the sensitivity to adversarial influence on the cooperative agents is undesirable in cases where security is an issue.

3. CONSENSUS PROTOCOL

This section introduces the Adversarially Robust Consensus Protocol (ARC-P), which is robust to adversaries in complete networks whenever there are more cooperative agents than adversaries, or $n > 2F$. The main idea of the protocol is for each agent to sort the state values and then filter (remove) the F largest and F smallest values so that the remaining values lie within the range of cooperative states. The state of the given agent i is then subtracted from each of the remaining values to form $m = n - 2F$ relative state values. A relative state value is negative if the state of agent i is greater than the filtered state value and nonnegative otherwise. The rate of change of the state of agent i is then the sum of these m relative state values. The result is that the state of agent i increases (decreases) whenever it is smaller (larger) than the average of the m filtered values, and remains constant if it is equal. Intuitively, this process should force the cooperative agents to converge to the average of the filtered values. In the extreme case, whenever $n = 2F + 1$, only the median of the state values remains, and therefore, the cooperative agents' states are driven toward the median of the state values. Before stating the protocol, we require some definitions.

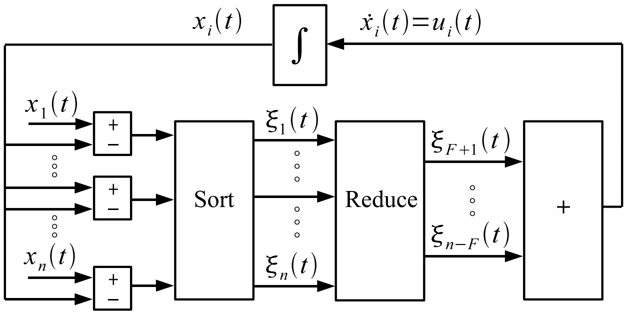


Figure 2: Synchronous data flow model of ARC-P for agent i.

Definition 1. *Let $m = n - 2F$, and denote the elements of a vector $x \in \mathbb{R}^n$ by x_i, $i = 1, 2, \ldots, n$. Then:*

1. *The concatenation function, $\chi_{p,q} \colon \mathbb{R}^p \times \mathbb{R}^q \to \mathbb{R}^{p+q}$, is defined by*[1]
$$\chi_{p,q}(y, z) = \begin{bmatrix} y \\ z \end{bmatrix}; \tag{5}$$
2. *The (ascending) sorting function on n elements, $\rho_n \colon \mathbb{R}^n \to \mathbb{R}^n$, is defined by $\xi = \rho_n(x)$ such that ξ is a permutation of x satisfying*
$$\xi_1 \leq \xi_2 \leq \cdots \leq \xi_n; \tag{6}$$
3. *The reduce function with respect to $F \in \mathbb{Z}^+$ is defined by $r_F \colon \mathbb{R}^n \to \mathbb{R}^{n-2F}$, $n > 2F$, satisfying*
$$r_F(\xi) = [\xi_{F+1}\ \xi_{F+2}\ \ldots\ \xi_{n-F}]^T; \tag{7}$$
4. *The sum function, $s \colon \mathbb{R}^m \to \mathbb{R}$, is defined by*
$$s(x) = \sum_{i=1}^{m} x_i; \tag{8}$$
5. *The composition of the concatenation, sorting, reduce, and sum functions is defined by $\phi \colon \mathbb{R}^p \times \mathbb{R}^q \to \mathbb{R}$, satisfying for all $(y, z) \in \mathbb{R}^p \times \mathbb{R}^q$,*
$$\phi(y, z) = (s \circ r_F \circ \rho \circ \chi_{p,q})(y, z). \tag{9}$$

The concatenation, sorting, and sum functions are defined in a natural way. The reduce function is intended to be composed with the sorting function – as in the definition of ϕ – so that the resulting operation removes the F smallest and F largest elements.

With these definitions, ARC-P calculates $u_i{=}f_i(x_c, x_a)$ for each cooperative agent $i \in \mathcal{V}_c$ by

$$f_i(x_c(t), x_a(t)) = -m x_i(t) + \phi(x_c(t), x_a(t)), \tag{10}$$

where $m = n - 2F$. Thus, to ensure $m \geq 1$, we require $n > 2F$.

Figure 2 shows the data flow model of ARC-P for cooperative agent i. In the figure, the state, $x_i(t)$, of the agent, whose dynamics are $\dot{x}_i(t) = u_i(t)$, is subtracted from each of the other states,

[1] The concatenation function $\chi_{p,q}$ is essentially the identity function on $\mathbb{R}^{p+q}$. It is used for notational reasons so that x_a can be treated as an uncertain input to the system of cooperative agents.

including itself. The resulting relative state values are sorted and then reduced by eliminating the largest and smallest F elements. Finally, the remaining elements are summed to produce the control input $u_i(t)$ to the integrator agent. It is straightforward to show that this order of operations is equivalent to (10). This implementation is most beneficial in cases where the relative state is sensed directly, so there is no need for a global coordinate system.

4. ANALYSIS

In this section, we analyze the continuity and convergence properties of ARC-P. First, we consider the Lipschitz continuity of the protocol, in order to ensure existence and uniqueness of solutions for all time $t > 0$, over all initial conditions $x_c(0) \in \mathbb{R}^p$, and for all adversarial trajectories, $x_a(t) \in \mathcal{C}$. Next, we show the agreement condition is satisfied and characterize the convergence rate to the agreement space. Then, we prove that the validity condition holds, thereby showing that ARC-P solves the adversarial agreement problem. Finally, we consider a uniform upper bound on convergence in order to obtain ϵ-approximate solutions to the adversarial agreement problem in finite time.

4.1 Lipschitz Continuity

We begin by recalling the definition of Lipschitz continuity. For the purposes of this paper, we restrict ourselves to Euclidean spaces.

DEFINITION 2. *Let $||\cdot||$ denote any norm defined on a Euclidean space, and let $g\colon \mathbb{R}^n \to \mathbb{R}^m$. Then g is Lipschitz continuous with Lipschitz constant L if the following condition holds for all $x, y \in \mathbb{R}^n$:*

$$||g(x) - g(y)|| \leq L||x - y||.$$

In order to show the Lipschitz continuity of ARC-P, we must show that the sorting function is Lipschitz continuous. First, we consider an interesting property of the sorting function; namely, given any two vectors, then the angle between the vectors will never increase by sorting the vectors. This result is then used to show Lipschitz continuity of the sorting function.

LEMMA 1. *Given $x, x_0 \in \mathbb{R}^n$ with $\xi = \rho_n(x)$ and $\xi_0 = \rho_n(x_0)$, then*

$$\xi^{\mathsf{T}}\xi_0 = \sum_{i=1}^{n} \xi_i \xi_{0_i} \geq \sum_{i=1}^{n} x_i x_{0_i} = x^{\mathsf{T}} x_0. \tag{11}$$

PROOF. We prove the result by induction on n. The base step ($n = 1$) is obvious (since $\xi = x$, $\xi_0 = x_0$). Now, suppose (11) is true for $1 \leq n \leq m$, and let $n = m+1$, with $x, x_0, \xi, \xi_0 \in \mathbb{R}^{m+1}$. Let j (and k) denote the index of the element with minimum value in x (x_0). If there are multiple elements with minimum values in either vector, arbitrarily fix the index to correspond to one of the minimum values. There are two cases: $j \neq k$ and $j = k$.

Case 1, $j \neq k$: Swap the elements x_j and x_k in x so that the minimum values of x and x_0 occur in the same index (k in this case). Remove the k^{th} element from each vector and denote the resulting vectors by $x', x_0' \in \mathbb{R}^m$, and their corresponding sorted versions by ξ' and ξ_0' respectively. Then, by the inductive hypothesis

$$\sum_{i=1}^{m} \xi_i' \xi_{0_i}' \geq \sum_{i=1}^{m} x_i' x_{0_i}'. \tag{12}$$

But the terms in (12) are related to the terms in $x^{\mathsf{T}}x_0$ and $\xi^{\mathsf{T}}\xi_0$ as follows. For the right-hand side, the only elements altered in x are x_j and x_k, which have been swapped (with x_j removed), and only x_{0_k} has been removed from x_0, with no other changes to x_0. Thus, we have

$$\sum_{i=1}^{m} x_i' x_{0_i}' = \sum_{\substack{i=1\\ i\neq j,k}}^{m+1} x_i x_{0_i} + x_k x_{0_j}. \tag{13}$$

Similarly, for the left-hand side of (12), only one minimum value of each vector has been removed; therefore, the inner product of the resulting sorted vectors (${\xi'}^{\mathsf{T}}\xi_0'$) contain the same terms as $\xi^{\mathsf{T}}\xi_0$, except for the term $x_j x_{0_k} = \xi_1 \xi_{0_1}$. Hence,

$$\sum_{i=1}^{m} \xi_i' \xi_{0_i}' = \sum_{i=2}^{m+1} \xi_i \xi_{0_i}. \tag{14}$$

Substituting (13) and (14) into (12) and adding $x_j x_{0_k} = \xi_1 \xi_{0_1}$ to both sides of the inequality yields

$$\sum_{i=1}^{m+1} \xi_i \xi_{0_i} \geq \sum_{\substack{i=1\\ i\neq j,k}}^{m+1} x_i x_{0_i} + x_k x_{0_j} + x_j x_{0_k}. \tag{15}$$

Now, since $x_k \geq x_j$ and $x_{0_j} \geq x_{0_k}$, we have

$$\begin{aligned}(x_k - x_j)(x_{0_j} - x_{0_k}) &\geq 0\\ \implies x_k x_{0_j} + x_j x_{0_k} &\geq x_k x_{0_k} + x_j x_{0_j}.\end{aligned}$$

Finally, combining this with (15) produces the desired result

$$\sum_{i=1}^{m+1} \xi_i \xi_{0_i} \geq \sum_{i=1}^{m+1} x_i x_{0_i}, \tag{17}$$

which completes the inductive step.

Case 2, $j = k$: Fix x' and x_0' by removing the k^{th} element (the minimum value) of x and x_0, respectively. Then, (12) is true by the inductive hypothesis. Analogous to Case 1, (14) also holds. In this case, the right-hand side of (12) is given by

$$\sum_{i=1}^{m} x_i' x_{0_i}' = \sum_{\substack{i=1\\ i\neq k}}^{m+1} x_i x_{0_i}. \tag{18}$$

Substituting (14) and (18) into (12) and adding $x_k x_{0_k} = \xi_1 \xi_{0_1}$ to both sides of the inequality yields (17), which completes the inductive step and the proof. □

LEMMA 2. *The sorting function, $\xi = \rho_n(x) \in \mathbb{R}^n$, defined by (6), is a Lipschitz continuous function of $x \in \mathbb{R}^n$.*

PROOF. Fix $x, x_0 \in \mathbb{R}^n$ and let $\rho_n(x) = \xi$, $\rho_n(x_0) = \xi_0$. Then, using the norm preservation property of permutations and Lemma 1, we have

$$\begin{aligned}||\xi - \xi_0||_2 &= \left(\xi^{\mathsf{T}}\xi + \xi_0^{\mathsf{T}}\xi_0 - 2\xi^{\mathsf{T}}\xi_0\right)^{\frac{1}{2}}\\ &\leq \left(x^{\mathsf{T}}x + x_0^{\mathsf{T}}x_0 - 2x^{\mathsf{T}}x_0\right)^{\frac{1}{2}} = ||x - x_0||_2.\end{aligned}$$ □

THEOREM 1. *The function $f_c(x_c, x_a)$ defining the dynamics of the subsystem of cooperative agents (c.f. (1)), with f_i defined in (10), is Lipschitz continuous in x_c and x_a.*

PROOF. The concatenation, reduce, and sum functions are linear maps and are therefore Lipschitz continuous. The sorting function is Lipschitz continuous by Lemma 2. The result then follows since scalar multiplication is Lipschitz continuous and the composition and difference of Lipschitz functions result in a Lipschitz continuous function. □

Since $x_a(t) \in \mathcal{C}$ is a bounded continuous trajectory and f_c is Lipschitz continuous, the system (1) admits a solution $x_c(t)$ which is uniquely defined on $\mathbb{R}^+$ for all $x_c(0) \in \mathbb{R}^p$ and $x_a(t) \in \mathcal{C}$ [2].

COROLLARY 1. *The networked multi-agent system with $n > 2F$ and each cooperative agent's control protocol given by (10), has a unique solution for all $t \geq 0$, $x_c(0) \in \mathbb{R}^p$, and $x_a(t) \in \mathcal{C}$.*

4.2 Agreement

In this section, we prove the agreement condition for ARC-P and characterize the convergence rate to the agreement space.

THEOREM 2. *The networked multi-agent system with $n > 2F$ and each cooperative agent's control protocol given by (10), satisfies the agreement condition (2). Moreover, the convergence to the agreement space is exponential with rate $m = n - 2F$.*

PROOF. For each pair $i, j \in \mathcal{V}_c$, $i \neq j$, we define the pairwise error $e_{ij}(t) = x_i(t) - x_j(t)$. Since $n > 2F$, $\phi(x_c(t), x_a(t))$ is defined. Because the network is complete, $\phi(x_c(t), x_a(t))$ is identical for each agent. Therefore, the ϕ-terms cancel in the error dynamics of $\dot{e}_{ij}(t) = -me_{ij}(t)$. Now, define $e(t)$ as the column vector containing all $\binom{p}{2}$ pairwise errors of the form $e_{ij}(t)$. Clearly, $e = 0$ is equivalent to $x_c \in \mathcal{A}$. The error dynamics are then

$$\dot{e} = -me \implies e(t) = e(0)\mathrm{e}^{-mt},$$

which proves the agreement condition is satisfied and that the convergence is exponential with rate $m = n - 2F$. □

4.3 Validity

While the agreement condition follows directly from the symmetry provided by the complete network, the validity condition requires an invariant set argument, facilitated by some results from the theory of uncertain systems. As described in Section 2.1, we consider a decomposition of the multi-agent system into cooperative and adversarial agents which interact through sharing state information. The state information from the adversarial agents is viewed as an uncertain input to the subsystem of cooperative agents and may take values in the compact set $\mathcal{C}$. We begin with the definition of robustly positively invariant sets.

DEFINITION 3. *The set $\mathcal{S} \subset \mathbb{R}^p$ is robustly positively invariant for the system given by (1) if for all $x_c(0) \in \mathcal{S}$ and all $x_a(t) \in \mathcal{C}$ the solution is such that $x_c(t) \in \mathcal{S}$ for $t > 0$.*

In order to show that the validity condition holds, we first show that the hypercube $\mathcal{H}_0$, containing all of the initial values of the cooperative agents, is a robustly positively invariant set using an extension of Nagumo's Theorem for uncertain systems. Then we prove that the limit of the cooperative agents' states exists and therefore lies in this hypercube. For notational brevity, we denote

$$x_{\text{min}} = \min_{i \in \mathcal{V}_c} x_i(0) \text{ and } x_{\text{max}} = \max_{i \in \mathcal{V}_c} x_i(0).$$

LEMMA 3. *If each cooperative agent's control protocol is given by (10), then the hypercube $\mathcal{H}_0 = \mathcal{I}_0^p$ defined by*

$$\mathcal{H}_0 = \{y \in \mathbb{R}^p | x_{min} \leq y_i \leq x_{max}\},$$

is robustly positively invariant for the system (1).

PROOF. First we require a definition. For any compact and convex set $\mathcal{S} \subset \mathbb{R}^p$, the tangent cone to $\mathcal{S}$ in y is the set

$$\mathcal{T}_{\mathcal{S}}(y) = \{z \in \mathbb{R}^p | \lim_{h \to 0} \frac{\text{dist}(y + hz, \mathcal{S})}{h} = 0\},$$

where $\text{dist}(y, \mathcal{S}) = \inf_{x \in \mathcal{S}} ||y - x||_2$. Since $\mathcal{H}_0$ is closed and convex, an extension to Nagumo's Theorem presented in [3, p.106] states that $\mathcal{H}_0$ is robustly positively invariant if and only if

$$f_c(y, x_a) \in \mathcal{T}_{\mathcal{H}_0}(y), \ \forall y \in \mathcal{H}_0 \text{ and } x_a \in \mathcal{C}.$$

For interior points y in $\mathcal{H}_0$, we have $\mathcal{T}_{\mathcal{H}_0}(y) = \mathbb{R}^p$, so we only need to check the boundary of $\mathcal{H}_0$. The boundary, $\partial\mathcal{H}_0$, is given by

$$\partial\mathcal{H}_0 = \{y \in \mathcal{H}_0 | \exists i \in \{1, 2, \ldots, p\} \text{ s.t. } y_i \in \{x_{\text{min}}, x_{\text{max}}\}\}.$$

In other words, points on the boundary have at least one component that is either a minimum or maximum of the initial values.

Fix $y \in \partial\mathcal{H}_0$ and let $\mathcal{I}_{\text{min}}, \mathcal{I}_{\text{max}} \subseteq \{1, 2, \ldots, p\}$ denote the sets of indices such that

$$j \in \mathcal{I}_{\text{min}} \Leftrightarrow y_j = x_{\text{min}} \text{ and } k \in \mathcal{I}_{\text{max}} \Leftrightarrow y_k = x_{\text{max}}.$$

Since $y \in \partial\mathcal{H}_0$, at least one of these index sets is nonempty. Let e_j denote the j-th canonical basis vector. From the geometry of the hypercube, it is sufficient for the following to hold:

$$e_j^\top f_c(y, x_a) \geq 0 \quad \forall j \in \mathcal{I}_{\text{min}}, x_a(t) \in \mathcal{C},$$
$$e_k^\top f_c(y, x_a) \leq 0 \quad \forall k \in \mathcal{I}_{\text{max}}, x_a(t) \in \mathcal{C}.$$

It can be shown that each component $j \in \{1, 2, \ldots, m\}$ of $\zeta \triangleq (r_F \circ \rho_n \circ \chi_{p,q})(y, x_a)$ satisfies

$$x_{\text{min}} \leq \zeta_j \leq x_{\text{max}}, \ \forall x_a(t) \in \mathcal{C}.$$

Indeed, otherwise $y \notin \mathcal{H}_0$. Therefore, we have that for all $x_a(t) \in \mathcal{C}$, $j \in \mathcal{I}_{\text{min}}$, and $k \in \mathcal{I}_{\text{max}}$,

$$e_j^\top f_c(y, x_a) = -my_j + \sum_{i=1}^{m} \zeta_i = -mx_{\text{min}} + \sum_{i=1}^{m} \zeta_i \geq 0,$$
$$e_k^\top f_c(y, x_a) = -my_k + \sum_{i=1}^{m} \zeta_i = -mx_{\text{max}} + \sum_{i=1}^{m} \zeta_i \leq 0. \quad \square$$

THEOREM 3. *The networked multi-agent system with $n > 2F$ and each cooperative agent's control protocol given by (10), satisfies the validity condition (3).*

PROOF. Consider $\psi(t) = \max_{i \in \mathcal{V}_c}(x_i(t))$. Define for each $j \in \mathcal{V}_c$,

$$\mathcal{I}_j = \{t \geq 0 \mid x_j(t) = \psi(t)\}.$$

Then, let $\mathcal{S} = \{j \in \mathcal{V}_c | \mathcal{I}_j \neq \emptyset\}$. We claim that elements of $\mathcal{S}$ satisfy the following property: for $i, j \in \mathcal{S}$, $x_i(t) \equiv x_j(t)$, for all $t \geq 0$. To show this, fix $i, j \in \mathcal{S}$, and consider the errors $e_{ij}(t)$ from the proof of Theorem 2. If $x_i(0) > x_j(0)$, then

$$e_{ij}(t) = e_{ij}(0)\mathrm{e}^{-mt} > 0, \quad \forall t \geq 0.$$

This contradicts the fact that $j \in \mathcal{S}$. By symmetry, we cannot have $x_j(0) > x_i(0)$. Therefore, $x_i(0) = x_j(0)$, and $e_{ij}(t) \equiv 0$, which implies $x_i(t) \equiv x_j(t)$, for all $t \geq 0$.

Therefore, $\psi(t)$ uniquely describes the positive trajectory of the subset of cooperative agents with initial value x_{max}. Now, since $\phi(x_c(t), x_a(t)) \leq m\psi(t)$, $\psi(t)$ is nonincreasing. Furthermore, since $\psi(t)$ must remain in $\mathcal{I}_0$ by Lemma 3, it is bounded below by x_{min}; thus $\lim_{t \to \infty} \psi(t) \in \mathcal{I}_0$ exists. Finally, by Theorem 2, all of the states of the cooperative agents converge to $\psi(t)$ and therefore, we have $\lim_{t \to \infty} \psi(t) = \lim_{t \to \infty} x_i(t)$, for all $i \in \mathcal{V}_c$. Since this is independent of $x_a(t)$, the validity condition is satisfied. □

By Theorem 2, ARC-P satisfies the agreement condition. By Theorem 3, ARC-P satisfies the validity condition. Therefore, ARC-P solves the adversarial agreement problem.

THEOREM 4. *The networked multi-agent system with $n > 2F$ and each cooperative agent's control protocol given by (10), solves the adversarial agreement problem.*

4.4 Finite Termination

In this section, we derive an upper bound on the performance of ARC-P in order to terminate in finite time while ensuring an ϵ-approximate solution to the adversarial agreement problem. By Theorem 2, the rate of convergence is exponential with rate m. But, the upper bound on convergence can be made precise.

COROLLARY 2. *Consider a networked multi-agent system with $n > 2F$ and each cooperative agent's control protocol given by (10). Define $\beta = \max_{i \in \mathcal{V}} x_i(0) - \min_{i \in \mathcal{V}} x_i(0)$. Then,*

$$\max_{i,j \in \mathcal{V}_c} e_{ij}(t) \leq \beta \mathsf{e}^{-mt}, \forall t \geq 0.$$

PROOF. For all $i, j \in \mathcal{V}_c$,

$$e_{ij}(t) = e_{ij}(0)\mathsf{e}^{-mt} \leq \beta \mathsf{e}^{-mt}, \forall t \geq 0. \qquad \square$$

Using Corollary 2, an ϵ-approximate solution to the adversarial agreement problem can be obtained in finite time. Specifically, to ensure that the maximum pairwise error between the states of cooperative agents is less than $\epsilon > 0$, Corollary 2 implies we may terminate at any time greater than $\frac{1}{m}|\log(\frac{\epsilon}{\beta})|$ (provided $\beta \neq 0$, in which case $x_c(0) \in \mathcal{A}$).

5. SIMULATIONS

To illustrate the robustness of ARC-P, we consider three examples in which a subset of the agents have been overtaken and redesigned with malicious intent, and a fourth example that illustrates the performance tradeoff incurred for the robustness to adversaries. The first scenario considers the case where two out of the eight agents are adversaries and their goal is to drive the consensus state of the cooperative agents to an unsafe set $\mathcal{U}$. In the second scenario, three of the agents have been redesigned as oscillators in order to force the cooperative agents to oscillate at the desired frequency. Finally, in the third scenario a single adversary in a large network tries to force the other agents to follow a sinusoidal trajectory in the unsafe set.

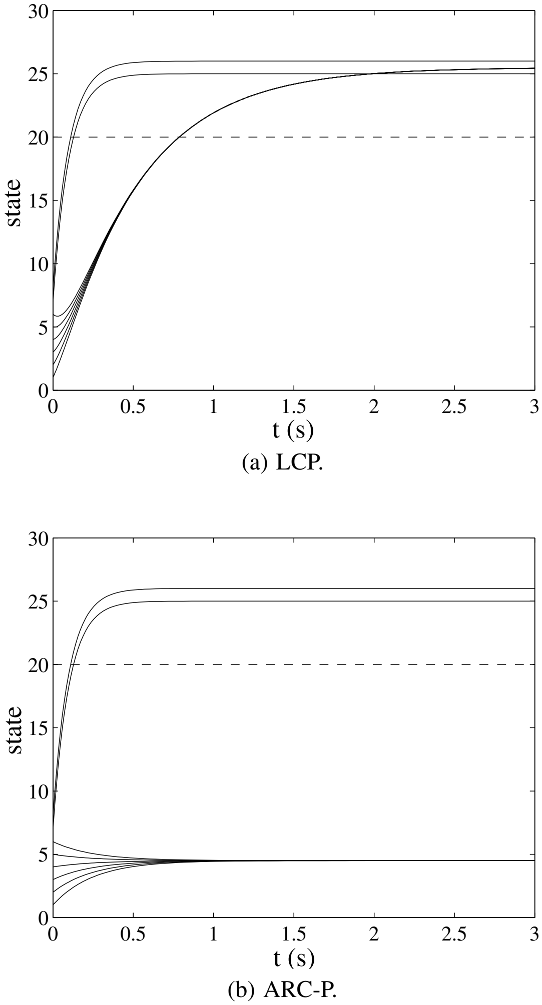
To motivate the need for a consensus protocol that is robust to adversaries, we compare ARC-P with LCP under the same conditions. It is shown that LCP achieves the agreement condition in spite of the behavior of the adversaries, but not the validity condition. For LCP, the states of the cooperative agents effectively converge to the average of the aversaries' trajectories. Thus, in all three scenarios, the adversaries are able to achieve their goal.

Example 1: Consider a multi-agent network with eight agents, each with unique identifier in $\{1, 2, \ldots, 8\}$, and with initial states equal to their identifier (e.g., for agent 1, $x_1(0) = 1$). Suppose that agents 7 and 8 have been compromised (i.e., $\mathcal{V}_a = \{7, 8\}$). The adversaries are redesigned with

$$\dot{x}_i = -10x_i + 10u_i, \; \forall i \in \mathcal{V}_a,$$

where the reference inputs u_i for the adversarial agents are $u_7 = 25$ and $u_8 = 26$. Therefore, the adversarial agents will converge exponentially to 25 and 26, respectively, with rate 10.

The goal of the adversaries is to drive the states of the cooperative agents into the unsafe set $\mathcal{U} = \{y \in \mathbb{R} | y \geq 20\}$. The results for LCP and ARC-P are shown in Figure 3. The adversaries are able to achieve their goal only with LCP. The cooperative agents equipped with ARC-P achieve both the agreement and validity conditions. Because both of the adversaries always have larger state values, the consensus process for the cooperative agents is unaffected and the final consensus state is the average of the $m = 4$ initial states of the agents filtered by $\phi(x_c, x_a)$; in this case, 4.5. Also, the rate of convergence is $m = 4$.



(a) LCP.

(b) ARC-P.

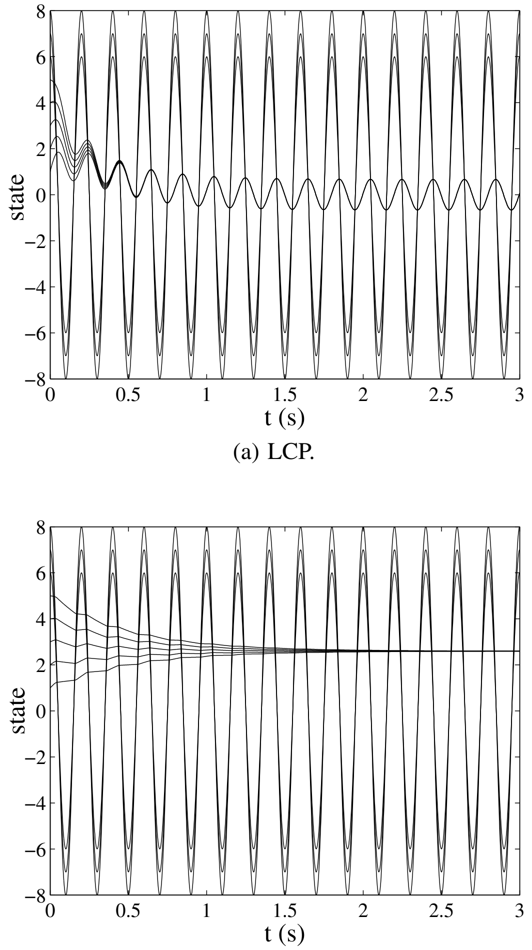
Figure 3: Adversaries try to drive agents to $\mathcal{U}$.

Example 2: Consider the same multi-agent network as Example 1, with the same initial conditions, but with agents 6, 7, and 8 as adversaries (i.e., $\mathcal{V}_a = \{6, 7, 8\}$). This time the adversaries' dynamics are designed as

$$\ddot{x}_i = -100\pi^2 x_i, \; \forall i \in \mathcal{V}_a.$$

Thus, they are oscillators with natural frequency 5 Hz and amplitude given by their initial state. The goal of the adversaries in this case is to force the cooperative agents' states to oscillate at 5 Hz.

The results for LCP and ARC-P are shown in Figure 4. As can be seen in Figure 4(a), the cooperative agents executing LCP synchronize and begin oscillating at 5 Hz, with a phase lag of 90° with respect to the adversaries. However, for the case of ARC-P, the cooperative agents achieve the agreement and validity conditions. As the adversarial agents move their states inside the range of the filter ϕ, the limit point for the cooperative agents is shifted, which can be seen in Figure 4(b) as a change in the shape of the exponential decay each time the adversaries move through this range.



(a) LCP.

(b) ARC-P.

Figure 4: Adversaries try to force agents to oscillate at 5 Hz.

This shifts the limit point from 3 to 2.6, without affecting the rate of consensus.

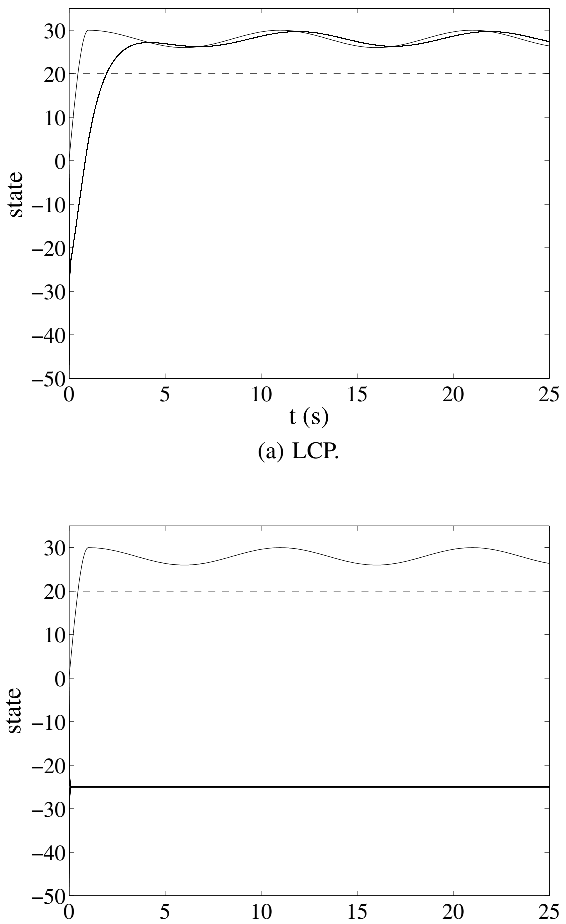
Example 3: Consider a multi-agent network with 51 agents, where only agent 51 is an adversary. The initial states of the cooperative agents are $x_1(0) = -1$, $x_2(0) = -2, \ldots, x_{50}(0) = -50$. The adversary is designed with time-varying dynamics given by the following expressions:

$$\begin{aligned} \ddot{x}_{51} &= -0.25\pi^2 x_{51} && \text{if } 0 \leq t < 1; \\ \dot{x}_{51} &= -0.4\pi \sin(0.2\pi(t-1)) && \text{if } t \geq 1; \end{aligned}$$

and initial conditions $x_{51}(0) = 0$, $\dot{x}_{51}(0) = 15\pi$. The resulting trajectory is

$$x_{51}(t) = \begin{cases} 30 \sin(0.5\pi t) & t < 1; \\ 2\cos(0.2\pi(t-1)) + 28 & t \geq 1. \end{cases}$$

The objective of the adversary is to bring the states of the cooperative agents into the unsafe set $\mathcal{U}$ (as in Example 1), and force them to oscillate at a frequency of 0.1 Hz. The results for LCP and ARC-P are shown in Figure 5. In this case, the convergence rates for LCP and ARC-P are 51 and 49, respectively, so the pairwise difference between the states of cooperative agents becomes negligible by 0.1 s (approximately five time constants) into the simulation. The result is that the trajectory of the cooperative agents appears to be a single curve in the figure. Clearly, the adversary is able to achieve its objective only with LCP. The consensus limit point for ARC-P is -25, i.e., the average of $x_1(0)$,. . . ,$x_{49}(0)$.



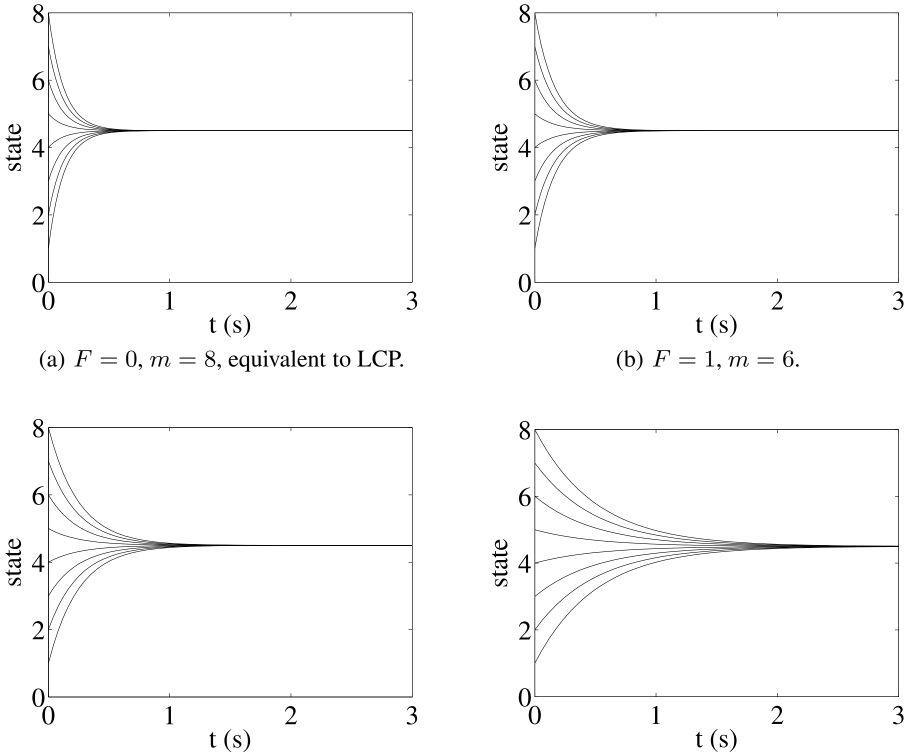
(a) LCP.

(b) ARC-P.

Figure 5: A single adversary tries to drive 50 cooperative agents to oscillate at 0.1 Hz in the unsafe set $\mathcal{U}$.

Example 4: We consider the performance tradeoff required for robustness to adversaries. For this purpose, consider the same multi-agent network of Example 1, but with no adversaries. As shown in Theorem 2, the rate of exponential convergence is $m = n - 2F$. The change in the rate of convergence with F is illustrated in Figure 6 for the eight agent network. Note that ARC-P reduces to LCP in the case $F = 0$. It is also important to note that although the limit point observed in Figure 6 is the same in all four cases, this would not be the case with asymmetries in the initial conditions.

By scaling ARC-P in (10) by the factor $\frac{n}{m}$, we may eliminate the tradeoff in performance for robustness to adversaries, and make ARC-P perform as well as LCP. However, LCP may also be scaled to improve its rate of convergence. Moreover, scaling ARC-P will incur a reduction in robustness to time delays as it does with LCP. Indeed, scaling LCP scales the largest eigenvalue of the Laplacian, which reduces the robustness to time delays [20]. Investigation of the robustness of ARC-P to time delays is left for future work.



(a) $F = 0, m = 8$, equivalent to LCP.

(b) $F = 1, m = 6$.

(c) $F = 2, m = 4$.

(d) $F = 3, m = 2$.

Figure 6: Performance tradeoff for robustness with ARC-P.

6. RELATED WORK AND DISCUSSION

Some of the earliest work on reaching consensus in the presence of adversaries can be found in [28] and [16], where the Byzantine agreement problem was first introduced. The notion of adversary described in this paper is similar to Byzantine failures, and has the same intent, i.e., to consider robustness to some sense of worst-case behavior. However, the restriction on continuity of the trajectory and the requirement to share true state information make our model more limited than the Byzantine fault model. Byzantine faulty processors are allowed to change their state to any valid state, and are allowed to send different messages to different processors. But within the environment of Euclidean space and with evolution in continuous-time, the restriction of continuity is sensible and even required in many physical situations. Furthermore, if the state information is obtained through sensing or broadcast communication, it is reasonable to assume the same state information is received by all agents (under the assumption of perfect sensors).

The Byzantine agreement problem has been generalized to allow approximate solutions for both synchronous and asynchronous systems whenever the values are arbitrary real numbers [6]. The filter ϕ of this work is modeled after the class of approximation functions considered in [6]. In contrast to [6], we construct ϕ on the topology of Euclidean space as opposed to using multisets. Moreover, the select function is not needed because the adversaries cannot hide their true state. This is why we are able to loosen the constraint on the ratio of adversaries to $n \geq 2F + 1$ (from $n \geq 3F + 1$). An interesting consequence of the results presented here is that the class of approximation functions considered in [6] are Lipschitz continuous (the proof of the Lipschitz continuity of the select function follows because the select function is linear).

More recently, there has been work in mobile robotics, which views coordination problems from a distributed computing perspective. The goal of the research is to characterize the weakest set of assumptions required for achieving a certain coordinated task in finite time [29]. One of the tasks considered is for a group of robots to gather at a single point in space (i.e., rendezvous) in finite time [32]. In order to consider the weakest assumptions on the capabilities of the robots, it is common to assume that the robots are indistinguishable, have different local coordinate frames, and are oblivious (which means they do not remember past observations or computations performed in previous steps). In some cases, it is assumed that all robots are able to obtain the exact position of all other robots, which is similar to the case considered in this paper.

The robot gathering problem has also been studied in the presence of faults [1], including Byzantine faults [4]. From a distributed computing perspective, the issue of stability of the robot dynamics is ignored, and the evolution of the robots occur in computational cycles (e.g., the *Wait-Look-Compute-Move* cycle) [29]. The common computational models used are the ATOM model [32], where the full cycle is executed instantaneously and atomically, and the CORDA model [29], where each stage requires a (nonzero) finite amount of time to execute any given stage, and any non-null move action will result in a (nonzero) finite distance moved.

The gathering problem with Byzantine failures shares some similarities with the adversarial agreement problem. As mentioned above, in some cases all-to-all sharing of state information is assumed [4]. Also, the cooperative agents are oblivious, which is also the case in our work because the control input is static. Additionally, in ARC-P the cooperative agents are indistinguishable, although this is not explicitly required in our model.There are also some key differences between the gathering and adversarial agreement problems. In our model, it is implicitly assumed there is a global inertial coordinate frame, unless the relative states are the quantities sensed. Also, in the gathering problem, consensus to a point must be achieved in finite time, as opposed to the asymptotic requirement of the adversarial agreement problem. However, because the dynamics of the robots are not considered, stability is assumed, whereas in the adversarial agreement problem, the dynamics of the individual agent and of the subsystem of cooperative agents is fundamental. A consequence of modeling the dynamics as ODEs is that the system model is synchronous. This differs from the gathering problem with Byzantine faults, where the system is often assumed to be asynchronous [4].

Another closely related research problem is the issue of security of consensus in multi-robot systems [9]. A distributed intrusion detection system (IDS) has been developed using a hybrid model of robotic agents that monitors neighboring agents to detect non-cooperative agents using only local information [9]. The distributed IDS has been extended to deal with the case where some of the monitors provide false information [8], and improved by using past information [7]. The distributed IDS approach differs from ARC-P because the agents executing ARC-P are oblivious, so they do not require past information for robustness to adversaries. Moreover, the approximation functions can be computed in linear time, so they are very efficient. However, the adversarial agreement problem is not concerned with issues such as collision avoidance, which makes the agents executing ARC-P susceptible in this case. On the other hand, avoiding collisions with misbehaving agents has been considered for the distributed IDS [8].

Also, the issue of security of linear consensus networks has been studied. In [25], the issue of a single malicious agent is considered. The same authors later extended the work to characterize the connectivity of the network required to tolerate misbehaving agents and non-colluding agents in [26]. A computationally expensive but exact algorithm was presented in [26] to detect and isolate up to k misbehaving agents in networks with connectivity at least $2k+1$. Additionally the structure of the entire network was necessary for the exact algorithm [26]. In [27], two approaches were considered to reduce the computational complexity and require only partial network information. The first assumes the network is comprised of weakly interconnected subcomponents and restricts the behavior of the misbehaving agents. The second imposes a hierarchical structure to detect and isolate the misbehaving agents. The problem considered in these papers requires detecting and isolating the misbehaving agents, and therefore results in more complex algorithms than ARC-P.

Another approach is considered in [31], where the feasibility of reaching consensus on any function of the initial states is considered in the presence of malicious agents. In this case, the linearity of the protocol is exploited to calculate the initial values exactly in at most n steps, where n is the number of nodes. Similar to our model for adversary, the malicious agents send the same information to all their neighbors. However, the malicious agents are modeled in discrete-time, so there is no continuity restriction.

Finally, [12] presents a framework for determining the robustness of distributed algorithms for discrete-time, synchronous algorithms on undirected graphs. The robustness of an algorithm is defined with respect to a fault model and is measured using a cost function. The cost function is a formal model of the cooperative task, and is defined on a domain consisting of all state and input trajectories, initial states, and environmental variables of the networked multi-agent system. The minimizer of the cost function should occur on a subset of the domain where the networked multi-agent system achieves the task. By extending the framework of [12] to continuous-time and by defining the cost function

$$C = \lim_{t\to\infty}\left(\sum_{i,j=1}^{n}(x_i(t)-x_j(t))^2 + \sum_{i=1}^{n}\mathrm{dist}(x_i(t),\mathcal{H}_0)^2\right),$$

for the adversarial agreement problem, it is straightforward to show using the results of this paper that ARC-P is worst-case robust to adversaries up to $F = \lfloor\frac{n-1}{2}\rfloor$ agents in complete networks. This shows that our interpretation of robustness is conformable to that of [12].

7. CONCLUSIONS AND FUTURE WORK

In this paper, we provide a general model for adversaries in Euclidean space and propose the adversarial agreement problem, a consensus problem in the context of adversaries. Then, we introduce the Adversarially Resilient Consensus Protocol (ARC-P) that combines ideas from distributed computing and control consensus protocols. We analyze the convergence properties of ARC-P, and show that it solves the adversarial agreement problem. Additionally, we show how to obtain approximate solutions to the problem in finite time. Then we provide several simulations to illustrate the resilient behavior of ARC-P to adversaries and the performance tradeoff required for the robustness to adversaries. Finally, we describe the related work and show how ARC-P is worst-case robust [12] to adversaries in complete networks whenever the ratio of adversaries to total agents is less than one-half.

Currently, we assume an "ideal" network, where all agents are able to obtain instantaneous, real-valued state information from every other agent. This assumption isolates the network uncertainties from the issue of adversaries. In future work, we would like to relax these idealized assumptions by simultaneously considering adversaries with other network uncertainties, such as time delays, packet loss, channel noise, and quantization. Also, for the protocol to be more useful, we intend to generalize the protocol for arbitrary network topologies by using a local bound on the number of adversaries amongst each agent's neighbors.

8. ACKNOWLEDGMENTS

The authors would like to thank the reviewers for their comments, which have improved the paper. This work is supported in part by the National Science Foundation (CNS-1035655, CCF-0820088), the U.S. Army Research Office (ARO W911NF-10-1-0005), and Lockheed Martin. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the U.S. Government.

9. REFERENCES

[1] N. Agmon and D. Peleg. Fault-tolerant gathering algorithms for autonomous mobile robots. *SIAM J. Comput.*, 36(1):56–82, 2006.

[2] F. Blanchini. Set invariance in control. *Automatica*, 35(11):1747 – 1767, 1999.

[3] F. Blanchini and S. Miani. *Set-Theoretic Methods in Control*. Birkhauser, Boston, Massachusetts, 2008.

[4] Z. Bouzid, M. Gradinariu Potop-Butucaru, and S. Tixeuil. Byzantine convergence in robot networks: The price of asynchrony. In *Int. Conf. on Principles of Distributed Systems*, pages 54–70, December 2009.

[5] M. H. DeGroot. Reaching a consensus. *Journal of the American Statistical Association*, 69(345):118–121, 1974.

[6] D. Dolev, N. A. Lynch, S. S. Pinter, E. W. Stark, and W. E. Weihl. Reaching approximate agreement in the presence of faults. *Journal of the ACM*, 33(3):499 – 516, 1986.

[7] A. Fagiolini, F. Babboni, and A. Bicchi. Dynamic distributed intrusion detection for secure multi-robot systems. In *Int. Conf. of Robotics and Aut.*, pages 2723 – 2728, May 2009.

[8] A. Fagiolini, A. Bicchi, G. Dini, and I. Savino. Tolerating malicious monitors in detecting misbehaving robots. In *IEEE Int. Workshop on Safety, Security, and Rescue Robotics*, pages 108 – 114, October 2008.

[9] A. Fagiolini, M. Pellinacci, M. Valenti, G., G. Dini, and A. Bicchi. Consensus-based distributed intrusion detection for multi-robot systems. In *Int. Conf. on Robotics and Aut.*, pages 120 – 127, May 2008.

[10] J. A. Fax and R. M. Murray. Information flow and cooperative control of vehicle formations. *IEEE Trans. on Aut. Control*, 49(9):1465 – 1476, 2004.

[11] S. Gilbert, N. Lynch, S. Mitra, and T. Nolte. Self-stabilizing robot formations over unreliable networks. *ACM Trans. Auton. Adapt. Syst.*, 4(3):1–29, 2009.

[12] V. Gupta, C. Langbort, and R. Murray. On the robustness of distributed algorithms. In *IEEE Conf. on Decision and Control*, December 2006.

[13] P. Hovareshti, J. Baras, and V. Gupta. Average consensus over small world networks: A probabilistic framework. In *IEEE Conf. on Decision and Control*, December 2008.

[14] M. Huang and J. Manton. Stochastic Lyapunov analysis for consensus algorithms with noisy measurements. In *American Control Conf.*, July 2007.

[15] Q. Hui, W. Haddad, and S. Bhat. On robust control algorithms for nonlinear network consensus protocols. *Int. J. Robust Nonlinear Control*, 20(3):269 – 284, 2010.

[16] L. Lamport, R. Shostak, and M. Pease. The Byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(2):382–401, 1982.

[17] D. Lee and M. Spong. Agreement with non-uniform information delays. In *American Control Conf.*, pages 756 – 761, June 2006.

[18] N. A. Lynch. *Distributed Algorithms*. Morgan Kaufmann Publishers Inc., San Francisco, California, 1997.

[19] M. Mesbahi and M. Egerstedt. *Graph Theoretic Methods in Multiagent Networks*. Princeton University Press, Princeton, New Jersey, 2010.

[20] R. Olfati-Saber. Ultrafast consensus in small-world networks. In *American Control Conf.*, pages 2371 – 2378, June 2005.

[21] R. Olfati-Saber, J. A. Fax, and R. M. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, 2007.

[22] R. Olfati-Saber, E. Franco, E. Frazzoli, and J. Shamma. Belief consensus and distributed hypothesis testing in sensor networks. In *Networked Embedded Sensing and Control*, volume 331 of *Lecture Notes in Control and Information Sciences*, pages 169–182. Springer Berlin / Heidelberg, 2006.

[23] R. Olfati-Saber and R. M. Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Trans. on Aut. Control*, 49(9):1520 – 1533, September 2004.

[24] L. Pallottino, V. G. Scordio, E. Frazzoli, and A. Bicchi. Decentralized cooperative policy for conflict resolution in multi-vehicle systems. *IEEE Trans. on Robotics*, 23(6):1170–1183, 2007.

[25] F. Pasqualetti, A. Bicchi, and F. Bullo. Distributed intrusion detection for secure consensus computations. In *IEEE Conf. on Decision and Control*, pages 5594 –5599, December 2007.

[26] F. Pasqualetti, A. Bicchi, and F. Bullo. On the security of linear consensus networks. In *IEEE Conf. on Decision and Control*, December 2009.

[27] F. Pasqualetti, R. Carli, A. Bicchi, and F. Bullo. Identifying cyber attacks via local model information. In *IEEE Conf. on Decision and Control*, December 2010.

[28] M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *J. ACM*, 27(2):228–234, 1980.

[29] G. Prencipe. CORDA: Distributed coordination of a set of autonomous mobile robots. In *Proc. 4th European Research Seminar on Advances in Distributed Systems*, pages 185–190, May 2001.

[30] D. Spanos, R. Olfati-Saber, and R. Murray. Approximate distributed Kalman filtering in sensor networks with quantifiable performance. In *4th Int. Symp. on Information Processing in Sensor Networks*, pages 133 – 139, April 2005.

[31] S. Sundaram and C. Hadjicostis. Distributed function calculation via linear iterations in the presence of malicious agents; part II: Overcoming malicious behavior. In *American Control Conf.*, pages 1356 –1361, June 2008.

[32] I. Suzuki and M. Yamashita. Distributed anonymous mobile robots: Formation of geometric patterns. *SIAM Journal on Computing*, 28:1347–1363, 1999.

[33] H. Tanner, G. Pappas, and V. Kumar. Leader-to-formation stability. *IEEE Trans. on Robotics and Automation*, 20(3):443 – 455, 2004.

[34] S. Vanka, V. Gupta, and M. Haenggi. Power-delay analysis of consensus algorithms on wireless networks with interference. *Int. J. Syst., Control Commun.*, 2(1/2/3):256–274, 2010.

[35] W. Wang and J.-J. Slotine. Contraction analysis of time-delayed communications and group cooperation. *IEEE Trans. on Aut. Control*, 51(4):712–717, April 2006.

[36] L. Xiao, S. Boyd, and S. Lall. A scheme for robust distributed sensor fusion based on average consensus. In *4th Int. Symp. on Information Processing in Sensor Networks*, pages 63–70, April 2005.