Consensus of Multiagent Systems Via Asynchronous Cloud Communication

Sean L. Bowman, *Student Member, IEEE*, Cameron Nowzari, *Member, IEEE*, and George J. Pappas, *Fellow, IEEE*

***Abstract*—In this paper, we study a multiagent consensus problem in which agents are only able to communicate with each other intermittently through a cloud server. To reduce the amount of required communication, we develop a self-triggered algorithm that allows agents to communicate with the cloud only when necessary rather than at a fixed period. Unlike the vast majority of similar works that propose distributed event- and/or self-triggered control laws, this paper does not assume agents can be "listening" continuously. In other words, when an event is triggered by one agent, neighboring agents will not be aware of this until the next time they establish communication with the cloud themselves. Using a notion of "promises" about future control inputs, agents are able to keep track of higher quality estimates about their neighbors allowing them to stay disconnected from the cloud for longer periods of time while still guaranteeing a positive contribution to the global task. We prove that our self-triggered coordination algorithm guarantees that the system asymptotically reaches the set of desired states. Simulations illustrate our results.**

***Index Terms*—Algorithm design and analysis, convergence, Lyapunov methods, multi-agent systems, multi-robot systems.**

I. Introduction

THIS PAPER considers a multiagent consensus problem where agents can only communicate with one another indirectly through the use of a central base station or "cloud." Small connected household devices that require communication and coordination with each other are becoming increasingly prevalent (the "Internet of Things"). To reduce both power consumption and bandwidth requirements for these small, low-power devices, it is ideal that they communicate as infrequently as possible with the cloud server. For instance, one can imagine a number of devices trying to coordinate through asynchronous communication with a dedicated cloud (e.g., email) server. In this setting, a device can only receive and send messages while connected to the server; however, being connected to the server at all times is a waste of energy and wireless resources. In this paper, we present a method to facilitate the coordination of a number of agents through a cloud server that guarantees the completion of a global task while reducing the number of communications required and without the need for a device to continuously be in communication with the cloud server.

Specifically, we consider the concrete problem of driving a set of agents to consensus. Consensus is a multiagent coordination problem that is, despite its simplicity, applicable to numerous networking areas; for example, formation control, distributed estimation, and clock synchronization can all be directly or indirectly solved by applications of consensus algorithms [1]. Many groups of researchers have looked at this problem; however, the vast majority of related works allow for an agent to broadcast to its neighbors at any desired time [1], which requires each agent to be "listening" at all times. The proliferation of small devices with strict power consumption requirements that are unable to constantly have their radios turned ON renders these approaches inadequate. Instead, we seek a lower power solution that only requires communication at a limited discrete set of times.

Each time an agent communicates with the base server, it must determine the next communication time as well as the control law to use while disconnected in order to achieve some desired global task based only on information available on the server at that moment. In this paper, we are interested in designing a self-triggered coordination algorithm in which agents can autonomously schedule the next time to communicate with the cloud based on currently available information.

Literature review: In the context of the multiagent coordination problem in general, the literature is extensive [2], [3]. In our specific problem of multiagent consensus, Olfati-Saber and Murray [4] introduce a continuous-time law that guarantees consensus convergence on undirected as well as weight-balanced digraphs. However, the majority of these works assume agents can continuously, or at least periodically, obtain information about their neighbors. Instead, when communication is expensive, as in our case, we wish to minimize the number of times communication is necessary.

A useful tool for determining communication times in this manner is event-triggered control, where an algorithm is designed to tune controller executions to the state evolution of a given system, see, e.g., [5] and [6]. In particular, event-triggered

Manuscript received March 26, 2019; revised July 15, 2019; accepted July 20, 2019. Date of publication August 15, 2019; date of current version June 12, 2020. Recommended by Associate Editor S. L. Smith. *(Corresponding author: Sean L. Bowman.)*

S. L. Bowman is with the Computer and Information Science Department, University of Pennsylvania, Philadelphia, PA 19104 USA (e-mail: seanbow@seas.upenn.edu).

C. Nowzari is with the Electrical and Computer Engineering Department, George Mason University, Fairfax, VA 22030 USA (e-mail: cnowzari@gmu.edu).

G. J. Pappas is with the Electrical and Systems Engineering Department, University of Pennsylvania, Philadelphia, PA 19104 USA (e-mail: pappasg@seas.upenn.edu).

Digital Object Identifier 10.1109/TCNS.2019.2935626

2325-5870 © 2019 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 15:16:35 UTC from IEEE Xplore. Restrictions apply.

control has been successfully applied to multiagent systems with the goal of limiting computation and decision making to reduce overall communication, sensing, and/or actuation effort of the agents. In [7], the authors formulate a threshold on system error to determine when control signals need to be updated. In [8], the authors expand on this and determine a distributed threshold for a wireless control network, taking into account network errors such as communication delays and packet drops. Event-triggered ideas have also been applied to the acquisition of information. Several approaches [9]–[11] utilize periodically sampled data to re-evaluate the controller trigger. Zhong and Cassandras [12] additionally drop the need for periodic sampling, creating a distributed trigger to decide when to share data based only on local information.

Event-triggered approaches generally require the persistent monitoring of some triggering function as new information is being obtained. Unfortunately, this is not directly applicable to our setup because the agents only get new information when they communicate with the base station. Instead, self-triggered control [13]–[15] removes the need to continuously monitor the triggering function, instead requiring each agent to compute its next trigger time based solely on the information available at the previously triggered sample time.

The first to apply these ideas to consensus, Dimarogonas *et al.* [16], remove the need for continuous control by introducing an event-triggered rule to determine when an agent should update its control signal, however, still requiring continuous information about their neighbors. In [17], the authors further remove the need for continuous neighbor state information, creating a time-dependent triggering function to determine when to broadcast information. The algorithm presented in [18] similarly broadcasts information based on a state-dependent triggering function. Recently, these ideas have been extended from undirected graphs to arbitrary directed ones [11], [19].

A major drawback of all aforementioned works is that they require all agents to be "listening," or available to receive information, at all times. Specifically, when an agent decides to broadcast information to its neighbors, it is assumed that all neighbors in the communication graph are able to instantaneously receive that information. Instead, we are interested in a situation where when an agent is disconnected from the cloud it is incapable of communicating with other agents.

The authors in [20] consider the problem of multiagent coverage control and develop a cloud-supported, event-triggered algorithm to assign coverage regions. In [21], the authors study a similar problem to the one we consider here but in the context of coordination of autonomous underwater vehicles (AUVs) that are unable to communicate while submerged. The authors develop an event-triggered solution in which all AUVs must surface together at the same time. Instead, we are interested in a strategy in which devices can autonomously communicate asynchronously while still guaranteeing a desired stability property. This problem has recently been looked at in [22]–[25], where the authors utilize event- and self-triggered coordination strategies to determine when the AUVs should resurface. In [22], a time-dependent triggering rule is developed that ensures practical convergence (in the presence of noise) of the whole system to the desired configuration. In [24], the authors present a similarly time-dependent triggering rule that allows agents to track a reference trajectory in the presence of noise, and in [25], they generalize this to second-order systems. Instead, the authors in [23] develop a state-dependent triggering rule with no explicit dependence on time; however, the self-triggered algorithm developed there is not guaranteed to avoid Zeno behaviors, which makes it an incomplete solution. In this paper, we incorporate ideas of promises from team-triggered control [26] to develop a state-dependent triggering rule that guarantees asymptotic convergence to consensus while ensuring that Zeno behavior is avoided.

Statement of contributions: Our main contribution is the development of a novel distributed team-triggered algorithm that combines ideas from self-triggered control with a notion of "promises." These promises allow agents to make better decisions since they have higher quality information about their neighbors in general. Our algorithm incorporates these promises into the state-dependent trigger to determine when they should communicate with the cloud. In contrast to [22] and [23], our algorithm uses a state-dependent triggering rule with no explicit dependence on time, no global parameters, and no possibility of Zeno behavior. The main drawback of the time-dependent triggering rule is the dependence on parameters $\sigma_0, \sigma_1, \lambda_0$, which greatly affect the performance (number of events and convergence speed) of the system and there is no good way to choose these *a priori*; i.e., depending on the initial condition, different values of $\sigma_0, \sigma_1, \lambda_0$ will perform better. Instead, the state-dependent triggering rule developed here is more naturally coupled with the current state of the system. In general, distributed event- and self-triggered algorithms are designed so that agents are *never* contributing negatively to the global task, generally defined by the evolution of a Lyapunov function V. Instead, our algorithm does not rely on this guarantee. More specifically, we actually allow an agent to be contributing negatively to the global task temporarily, as long as it is accounted for by its net contribution over time. Our algorithm guarantees the system converges asymptotically to consensus while ensuring that Zeno executions cannot occur. Finally, we illustrate our results through simulations.

An earlier version of this paper appeared in [27]. In contrast to [27], the algorithm presented here much more effectively uses all available information from the cloud. Second, it is both motivated more rigorously as an optimization problem and fully proven correct. Finally, our simulations are greatly expanded and show that the algorithm presented here compares very favorably to that in [27].

II. Problem Statement

We consider a system of N agents with single-integrator dynamics

$$\dot{x}_i(t) = u_i(t) \tag{1}$$

for all $i \in \{1, \ldots, N\}$, where we are interested in reaching a consensus configuration, i.e., where $\|x_i(t) - x_j(t)\| \to 0$ as $t \to \infty$ for all $i, j \in \{1, \ldots, N\}$. For simplicity, we consider

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 15:16:35 UTC from IEEE Xplore. Restrictions apply.

scalar states $x_i \in \mathbb{R}$, but these ideas are extendable to arbitrary dimensions.

Given a connected communication graph $\mathcal{G}$, it is well known [4] that the distributed continuous control law

$$u_i(t) = -\sum_{j \in \mathcal{N}_i} (x_i(t) - x_j(t)) \tag{2}$$

drives each agent of the system to asymptotically converge to the average of the agents' initial conditions. However, in order to be implementable, this control law requires each agent to continuously have information about its neighbors and continuously update its control law.

Several recent works have been aimed at relaxing these requirements [11], [17], [19], [28]. However, they all require agents to be "listening" continuously to their neighbors, i.e., when an event is triggered by one agent, its neighbors are immediately aware and can take action accordingly.

Unfortunately, as we assume here that agents are unable to perform any communication except at a discrete set of communication times, we cannot continuously detect neighboring events that occur. Instead, we assume that agents are only able to update their control signals when their own events are triggered (i.e., when they communicate with the base server). Let $\{t_i^\ell\}_{\ell \in \mathbb{Z}_{\geq 0}}$ be the sequence of times at which agent i communicates. Then, our algorithm is based on a piecewise constant implementation of the controller (2) given by

$$u_i^\star(t) = -\sum_{j \in \mathcal{N}_i} (x_i(t_i^\ell) - x_j(t_i^\ell)), \quad t \in [t_i^\ell, t_i^{\ell+1}). \tag{3}$$

Remark II.1: Later we will allow the control input $u_i(t)$ to change in a limited way on $[t_i^\ell, t_i^{\ell+1})$, but for now we assume that it is piecewise constant on the intervals $[t_i^\ell, t_i^{\ell+1})$. Motivation for and details behind changing the control while disconnected are discussed later in Section III-C. •

The purpose of this paper is to develop a self-triggered algorithm that determines how the sequence of times $\{t_i^\ell\}$ and control inputs $u_i(t)$ can be chosen such that the system converges to the desired consensus state. More specifically, each agent i at each communication time t_i^ℓ must determine the next time $t_i^{\ell+1}$ and control $u_i(t)$ only using information available on the cloud at that instant. The closed-loop system should then have trajectories such that $|x_i(t) - x_j(t)| \to 0$ as $t \to \infty$ for all $i, j \in \{1, \ldots, N\}$. We describe the cloud communication model next.

A. Cloud Communication Model

We assume that there exists a base station or "cloud" that agents are able to upload data to and download data from. This cloud can store any finite amount of data but can perform no computation. This assumption allows the computational load to be shared or off-loaded onto individual agents as necessary, allowing the system to share and take advantage of the available computational resources. This also results in a more scalable system limited only by the cloud's available memory, allowing the network to take advantage of computational resources introduced by any additional agents.

TABLE I
DATA STORED ON THE CLOUD FOR ALL AGENTS i AT ANY TIME t

t_i^{last}	Last time agent i communicated
t_i^{expire}	Control expiration time of agent i
t_i^{next}	Next time agent i will communicate
$x_i(t_i^{\text{last}})$	Last updated position of agent i
$u_i(t_i^{\text{last}})$	Last trajectory of agent i
$M_i(t_i^{\text{last}})$	Most recent control promise from agent i

At any given time $t \in [t_i^\ell, t_i^{\ell+1})$, the cloud stores the following information about agent i: the last time $t_i^{\text{last}}(t) = t_i^\ell$ that agent i communicated, the next time $t_i^{\text{next}}(t) = t_i^{\ell+1}$ that agent i is scheduled to communicate, the state $x_i(t_i^{\text{last}})$ of agent i when it last communicated, and the last control signal $u_i(t_i^{\text{last}})$ used by agent i. The server also contains a control expiration time $t_i^{\text{expire}} \leq t_i^{\text{next}}$ and a *promise* M_i for each agent i, which will be explained later in Section III. This information is summarized in Table I.

For simplicity, we assume that agents can download/upload information to/from the cloud instantaneously. Let t_i^ℓ be a time at which agent i communicates with the cloud. The communication link is established at time t_i^ℓ, and we immediately update $t_i^{\text{last}} = t_i^\ell$ and $x_i(t_i^\ell)$ based on agent is current position.

While the link is open, agent i downloads all the information in Table I for each neighbor $j \in \mathcal{N}_i$. Using this information, agent i (instantaneously) computes its control signal $u_i(t_i^\ell)$ and next communication time $t_i^{\ell+1}$ such that it knows it will make a net positive contribution to the consensus objective over the interval $[t_i^\ell, t_i^{\ell+1})$. Finally, before closing the communication link, agent i calculates a promise M_i bounding its future control inputs and uploads all data to the server.

Remark II.2: Because of the existence of a centralized cloud server, it may be tempting to ask why the communication graph $\mathcal{G}$ is not always the complete graph K_N. Note that the amount of computation an agent does under our algorithm is quadratic in the number of neighbors $|\mathcal{N}_i|$ (see Remark III.3). To ensure scalability for agents with limited computational capabilities as the number of agents in the network grows large, it may be necessary to force a more limited communication topology. Furthermore, especially with the increasing popularity of software defined networking, it is true that while any agent i may be *able* to communicate with any other agent j, it should be avoided whenever possible. •

Remark II.3: It may also be tempting to consider a cloud that is instead allowed to perform some computation and simply instructs each agent to travel to the computed centroid of all agents' positions. While this would save both computation and communication in situations where applicable, it loses the benefits of our distributed computation as outlined earlier in this section and in Remark III.3. Furthermore, our distributed consensus scheme allows for dynamically adding or removing agents to or from the system, is more robust to control and/or estimation disturbances, and is more easily extensible to other multiagent coordination problems, e.g., formation control. •

Problem 1: Given N agents with dynamics (1) and the communication model described in Section II-A, for each agent i, find an algorithm that prescribes when to communicate with the

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 15:16:35 UTC from IEEE Xplore. Restrictions apply.

cloud based on currently available information and a control input $u_i(t)$ used in between communications $t \in [t_i^\ell, t_i^{\ell+1})$, such that

$$|x_j(t) - x_i(t)| \to 0 \tag{4}$$

as $t \to \infty$ for all agents $i, j \in \{1, \dots, N\}$.

III. Distributed Trigger Design

Consider the objective function

$$V(x(t)) = \frac{1}{2} x^T(t) L x(t) \tag{5}$$

where L is the Laplacian of the connected communication graph $\mathcal{G}$. Note that $V(x) \geq 0$ for all $x \in \mathbb{R}^N$ and $V(x) = 0$ if and only if $x_i = x_j$ for all $i, j \in \{1, \dots, N\}$. Thus, the function $V(x)$ encodes the objective of the problem and we are interested in driving $V(x) \to 0$. For simplicity, we drop the explicit dependence on time when referring to time t.

Taking the derivative of V with respect to time, we have

$$\dot{V} = \dot{x}^T L x = -\sum_{i=1}^{N} \dot{x}_i \sum_{j \in \mathcal{N}_i} (x_j - x_i). \tag{6}$$

Let us split up $\dot{V} = \sum_{i=1}^N \dot{V}_i$, where

$$\dot{V}_i \triangleq -\dot{x}_i \sum_{j \in \mathcal{N}_i} (x_j - x_i). \tag{7}$$

Note that we have essentially distributed $\dot{V}$ in a way that clearly shows how each agent's motion directly contributes to the global objective, allowing us to write

$$V(x(t)) = V(x(0)) + \sum_{i=1}^{N} \int_0^t \dot{V}_i(x(\tau)) d\tau. \tag{8}$$

Remark III.1: The algorithm presented here is generalizable to other multiagent system architectures, e.g., higher-dimension states or directed graphs, through the use of other objective functions.

To ensure that each agent is positively contributing to the objective at all times, we now wish to design a self-triggered algorithm such that $\dot{V}_i(x(t)) \leq 0$ for all agents i at all times t. Thus, at time t_i^ℓ, agent i must determine $t_i^{\ell+1}$ and $u_i(t)$ such that $\dot{V}_i(t) \leq 0$ for all $t \in [t_i^\ell, t_i^{\ell+1})$.

While in the fully developed algorithm we will allow an agent to modify its control while disconnected, for now we assume that the control input is constant on the entire disconnected interval and defer the discussion of the control "expiration time" t_i^{expire}, its motivation, and (minor) modifications to the algorithm to Section III-C.

Note that given the information agent i downloaded from the server at time t_i^ℓ, it is able to exactly compute the state of a neighboring agent $j \in \mathcal{N}_i$ up to the time the neighbor j next communicates, t_j^{next}. For any $t \in [t_i^\ell, t_j^{\text{next}}]$

$$x_j(t) = x_j\left(t_j^{\text{last}}\right) + u_j\left(t_j^{\text{last}}\right)\left(t - t_j^{\text{last}}\right). \tag{9}$$

At time t_j^{next}, however, agent j autonomously updates its control signal in a way unknown to agent i, making it difficult to determine how agent i should move without communicating with the cloud. To remedy this, we borrow an idea of *promises* from team-triggered control [26]. Suppose that although we do not know $\dot{x}_j(t)$ exactly for $t > T_i$, we have access to some bound $M_j(t) > 0$ such that $|\dot{x}_j(t)| \leq M_j(t)$.

Using this information, we introduce the notion of agent js reachable set as determinable by agent i. For any $j \in \mathcal{N}_i$, let $R_j^i(t)$ be the set of states at time t within which agent i can determine that agent j must be in. For $t \leq t_j^{\text{next}}$, agent i is able to determine $x_j(t)$ exactly and so $R_j^i(t) = \{x_j(t)\}$ is a singleton containing agent js exact position. For $t > t_j^{\text{next}}$, as all agent i knows is a bound on agent js control law, R_j^i is a ball that grows at a rate determined by agent js promise M_j as

$$R_j^i(t) = \begin{cases} \{x_j(t_j^{\text{last}}) + u_j(t_j^{\text{last}})(t - t_j^{\text{last}})\} & t \leq t_j^{\text{next}} \\ B(x_j(t_j^{\text{next}}), M_j(t)(t - t_j^{\text{next}})) & \text{otherwise} \end{cases} \tag{10}$$

where $B(x, r)$ is a closed ball of radius r centered at x.

We can now express the latest time that agent i can still be sure, it is contributing positively to the objective.

Definition 1: $T_i^\star$ is the first time $T_i^\star \geq t_i^\ell$ after which agent i can no longer guarantee it is positively contributing to the objective, the solution to the following:

$$\begin{aligned} &\underset{t \geq t_i^\ell}{\text{infimum}} && t \\ &\text{subject to} && \max_{x(t) \in R^i(t)} \dot{V}_i(x(t)) > 0 \end{aligned} \tag{11}$$

where $R^i(t)$ is defined as the set of all states $x_j(t)$ for $j \in \mathcal{N}_i \cup \{i\}$ such that each $x_j(t)$ satisfies $x_j(t) \in R_j^i(t)$.

It is easy to compute the solution to (11) exactly given the structure of $R_j^i(t)$ given above. Let π_t be a sorted ordering on the next communication times of all of agent is neighbors, i.e., let $\pi_t : [|\mathcal{N}_i|] \to \mathcal{N}_i$ be a one-to-one function such that $t_{\pi_t(1)}^{\text{next}} \leq t_{\pi_t(2)}^{\text{next}} \leq \cdots \leq t_{\pi_t(|\mathcal{N}_i|)}^{\text{next}}$. We abuse notation slightly by additionally setting $t_{\pi(0)}^{\text{next}} = t_i^{\text{last}}$ and $t_{\pi(|\mathcal{N}_i|+1)}^{\text{next}} = \infty$ so the union of the intervals $[t_{\pi(k)}^{\text{next}}, t_{\pi(k+1)}^{\text{next}})$ for $k \in \{0, 1, \dots, |\mathcal{N}_i|\}$ covers all $t \geq t_i^{\text{last}}$.

Proposition 1: Let $\tau_i^{\star,(k)}$ be the solution to the following optimization:

$$\begin{aligned} &\underset{t}{\text{infimum}} && t \\ &\text{subject to} && t_{\pi(k)}^{\text{next}} \leq t \leq t_{\pi(k+1)}^{\text{next}} \\ &&& \sum_{m'=1}^{k} \alpha_{i\pi(m')}\left(t_{\pi(m')}^{\text{next}}\right) + \sum_{m=k+1}^{|\mathcal{N}_i|} \alpha_{i\pi(m)}\left(t_i^{\text{last}}\right) \\ &&& \quad + \sum_{m'=1}^{k} \left(t - t_{\pi(m)}^{\text{next}}\right) \gamma_{i\pi(m')}\left(t_i^{\text{last}}\right) \\ &&& \quad + \sum_{m=k+1}^{|\mathcal{N}_i|} \left(t - t_{\pi(m)}^{\text{next}}\right) \beta_{i\pi(m)}\left(t_i^{\text{last}}\right) > 0 \end{aligned} \tag{12}$$

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 15:16:35 UTC from IEEE Xplore. Restrictions apply.

where

$$\alpha_{ij}(t) \triangleq -u_i(t)(x_j(t) - x_i(t)) \tag{13}$$

$$\beta_{ij}(t) \triangleq -u_i(t)(u_j(t) - u_i(t)) \tag{14}$$

$$\gamma_{ij}(t) \triangleq |u_i(t)|M_j(t) + u_i(t)^2. \tag{15}$$

Then, the solution $T_i^\star$ to (11) can be computed exactly as

$$T_i^\star = \min\{\tau_i^{\star,(k)} \mid k \in \{0,\dots,|\mathcal{N}_i|\}\}. \tag{16}$$

Proof: See Appendix A. ■

It is additionally possible to allow agent i to remain disconnected for longer by allowing $\dot{V}_i$ to temporarily become positive, as long as we select $t_i^{\ell+1}$ such that the total contribution to the objective V on the interval $[t_i^\ell, t_i^{\ell+1})$

$$\Delta V_i^\ell \triangleq \int_{t_i^\ell}^{t_i^{\ell+1}} \dot{V}_i(\tau)d\tau \tag{17}$$

is nonpositive.

Definition 2: T_i^{total} is the first time $T_i^{\text{total}} \geq t_i^\ell$ after which agent i can no longer guarantee its total contribution over the submerged interval is positive, the solution to the following:

$$\begin{aligned} \underset{t \geq t_i^\ell}{\text{infimum}} \quad & t \\ \text{subject to} \quad & \max_{x(t)\in R^i(t)} \int_{t_i^{\text{last}}}^{t} \dot{V}_i(\tau)d\tau > 0. \end{aligned} \tag{18}$$

To compute T_i^{total}, we follow a similar approach.

Proposition 2: Let $\tau_i^{tot,(k)}$ be the solution to the following optimization:

$$\begin{aligned} \underset{t}{\text{infimum}} \quad & t \\ \text{subject to} \quad & t_{\pi(k)}^{\text{next}} \leq t \leq t_{\pi(k+1)}^{\text{next}} \\ & \sum_{m'=1}^{k} \Big[\alpha_{i\pi(m')}\left(t_i^{\text{last}}\right)\left(t_{\pi(m')}^{\text{next}} - t_i^{\text{last}}\right) \\ & \quad + \frac{1}{2}\beta_{i\pi(m')}\left(t_i^{\text{last}}\right)\left(t_{\pi(m')}^{\text{next}} - t_i^{\text{last}}\right)^2 \\ & \quad + \alpha_{i\pi(m')}\left(t_{\pi(m')}^{\text{next}}\right)\left(t - t_{\pi(m')}^{\text{next}}\right) \\ & \quad + \frac{1}{2}\gamma_{i\pi(m')}\left(t_i^{\text{last}}\right)\left(t - t_{\pi(m')}^{\text{next}}\right)^2\Big] \\ & + \sum_{m=k+1}^{|\mathcal{N}_i|} \Big[\alpha_{i\pi(m)}\left(t_i^{\text{last}}\right)\left(t - t_i^{\text{last}}\right) \\ & \quad + \frac{1}{2}\beta_{i\pi(m)}\left(t_i^{\text{last}}\right)\left(t - t_i^{\text{last}}\right)^2\Big] > 0. \end{aligned} \tag{19}$$

Then, the solution T_i^{total} to (18) can be computed as

$$T_i^{\text{total}} = \min\{\tau_i^{tot,(k)} \mid k \in \{0,\dots,|\mathcal{N}_i|\}\}. \tag{20}$$

Proof: See Appendix B. ■

Remark III.2: Although the constraints in Propositions 1 and 2 appear complex, note that they are linear or quadratic in t and so the infimums can be solved for easily. Consider a problem of the form

$$\begin{aligned} \underset{t}{\text{infimum}} \quad & t \\ \text{subject to} \quad & g(t) > 0 \\ & t_1 \leq t \leq t_2 \end{aligned} \tag{21}$$

where $g(t)$ is a polynomial in t.

Let $r_1 \leq r_2 \leq \cdots \leq r_K$ be the roots of g that lie in the interval (t_1, t_2), and let $r_0 = t_1$ and $r_{K+1} = t_2$. The solution $t^\star$ to (21) is the smallest r_i, $i = 0,\dots,K$, such that $g(r_i) \geq 0$ and $g(\frac{1}{2}(r_i + r_{i+1})) > 0$. If no such r_i exists, $t^\star = \infty$. •

Remark III.3: The computation of the constraint coefficients in (19) and (12) takes $O(|\mathcal{N}_i|)$ time, solving the optimization given the constraint coefficients takes $O(1)$ time, and in both cases $|\mathcal{N}_i| + 1$ such problems must be solved. Thus, computation of $T^\star$ and T^{total} both takes $O(|\mathcal{N}_i|^2)$ time. Note that $|\mathcal{N}_i|$ for some agent i does not necessarily increase proportionally with N; a random connected graph $\mathcal{G}$ with N vertices can have just $|\mathcal{N}_i| \propto \log N$ neighbors, resulting in an agent's computation taking $O(\log N)$ time, which is sublinear in N. •

Selecting $t^{\ell+1} = T_i^\star$ ensures that $\dot{V}_i < 0$ over the disconnected interval, ensuring that agent i is making progress toward the global objective at all t. Selecting $t^{\ell+1} = T_i^{\text{total}}$ introduces a tradeoff; while this time allows the agent to disable its radio for longer, as it allows some positive contribution to the objective function, overall progress toward consensus is slower. Thus, we propose a tuning parameter $\sigma_i \in [0,1]$, selecting a time $t_i^{\ell+1}$ such that $T_i^\star \leq t^{\ell+1} \leq T_i^{\text{total}}$

$$t_i^{\ell+1} = (1-\sigma_i)T_i^\star + \sigma_i T_i^{\text{total}}. \tag{22}$$

By continuity of $\dot{V}_i$ and the definitions of $T_i^\star$ and T_i^{total}, it is guaranteed for $t_i^{\ell+1} \in [T_i^\star, T_i^{\text{total}}]$ that we still have $\Delta V_i^\ell \leq 0$ [as defined in (17)]. Setting all σ_i near 0 allows faster convergence with more frequent communication, while σ_i near 1 results in slower convergence but less frequent communication.

A. Selecting Promises M_j

As it is not possible in general for agent i to bound a neighbor agent js future control inputs from past state and control information, instead each agent makes a *promise* M_i about its future control inputs each time it connects to the server. In the preceding section, we assumed that the bound $|\dot{x}_j(t)| \leq M_j(t)$ was satisfied at all times without describing how to make it so. Here, we describe how to codesign the control laws $u_i(t)$ and promises $M_i(t)$ to ensure that this actually holds at all times.

Let M_i^ℓ be the promise made by agent i at time t_i^ℓ. From the constraints in Propositions 1 and 2, it is clear that the smaller M_j is for any $j \in \mathcal{N}_i$, the longer agent i is able to remain disconnected. However, limiting the control too much below the ideal control (3) will slow convergence.

We consider a promise rule in which at time t_i^ℓ agent i sets its promise to be a function of $|u_i^\star(t_i^\ell)|$ as

$$M_i^\ell = f\left(\left|u_i^\star(t_i^\ell)\right|\right). \tag{23}$$

For example, $f(x) = cx$ provides a parameter c that effectively allows another tradeoff between convergence speed and communication frequency. Note, however, that this does not mean

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 15:16:35 UTC from IEEE Xplore. Restrictions apply.

agent i can use its ideal control law at all times; if the new desired input is greater in magnitude than a previous promise, to remain truthful to previous promises agent i must wait until all of its neighbors download the new promise before it can use its desired control input.

Let τ_{ij}^{ℓ} be the time that agent j sees agent is ℓth promise, i.e., $\tau_{ij}^{\ell} = t_j^{\text{next}}(t_i^{\ell})$. On the interval $[t_i^{\ell}, t_i^{\ell+1})$, agent i needs to guarantee that all promises M_i currently believed by $j \in \mathcal{N}_i$ are abided by.

Let $p_{ij}^{\text{last}}(t)$ be the index of the most recent promise by agent i that agent j is aware of at time t, i.e.,

$$p_{ij}^{\text{last}}(t) = \underset{\ell \,:\, \tau_{ij}^{\ell} \le t}{\arg\max}\, \tau_{ij}^{\ell} \tag{24}$$

and let $\mathcal{P}_i^{\ell}$ be the set of promise indices that agent i must abide by when submerging on $[t_i^{\ell}, t_i^{\ell+1})$, i.e.,

$$\mathcal{P}_i^{\ell} = \left\{\, p_{ij}^{\text{last}}(t) \mid j \in \mathcal{N}_i, t \in [t_i^{\ell}, t_i^{\ell+1}) \,\right\}. \tag{25}$$

To abide by all promises that agent is neighbors believe about its controls, then, it simply needs to bound its control input magnitude by

$$u_i^{\max}(t_i^{\ell}) = \min_{k \in \mathcal{P}_i^{\ell}} M_i^k. \tag{26}$$

With this bound, the actual control law used and uploaded by agent i on the interval $[t_i^{\ell}, t_i^{\ell+1})$ is given by bounding the ideal control magnitude by $u_i^{\max}(t_i^{\ell})$, or

$$u_i(t_i^{\ell}) = \begin{cases} u_i^{\star}(t_i^{\ell}) & \left|u_i^{\star}(t_i^{\ell})\right| \le u_i^{\max}(t_i^{\ell}) \\ u_i^{\max}(t_i^{\ell}) \frac{u_i^{\star}(t_i^{\ell})}{\left|u_i^{\star}(t_i^{\ell})\right|} & \text{otherwise.} \end{cases} \tag{27}$$

B. Maximum Submerged Time

The method presented, thus, far is almost complete; however, consider the case in which a subset of the communication graph has locally reached "consensus" while the system as a whole has not. If there is some agent i such that at agent is next communication time t_i^{ℓ} we have $x_i(t_i^{\ell}) = x_j(t_i^{\ell})$ for all $j \in \mathcal{N}_i$, and furthermore that $u_j(t_i^{\ell}) = 0$ for all $j \in \mathcal{N}_i$, then it would set its next triggering time to infinity. An examination of the constraints in (12) and (19) then reveals that the feasible set in both cases is equal to the empty set $\emptyset$. The times $T_i^{\star}$ and T_i^{total} will, thus, be chosen as $\inf \emptyset = \infty$.

To guarantee consensus in all situations, and as it is impossible for agent i to obtain information outside of its neighbors, it is necessary to introduce a maximum disconnected time $t_{\max}$. If an agent i computes a t_i^{ideal} such that $t_i^{\text{ideal}} - t_i^{\text{last}} > t_{\max}$, the agent instead chooses $t_i^{\text{next}} = t_i^{\text{last}} + t_{\max}$. This ensures that despite a region of the communication network being at local "consensus," no agent will effectively remove itself from the system and information will continue to propagate. While a large $t_{\max}$ could result in delayed consensus if part of the communication graph is at a local consensus, a too small $t_{\max}$ will result in unnecessary communication taking place; in simulation the local consensus situation is very rare and $t_{\max}$ played a small role in the communication time selection.

C. Avoiding Zeno Behavior

While the presented method of selecting times and control inputs guarantees convergence, it is susceptible to Zeno behavior, i.e., requiring some agent i to communicate an infinite number of times in a finite time period. To avoid this behavior, we introduce a fixed dwell time $T_i^{\text{dwell}} > 0$, and force each agent to remain disconnected for at least a duration of T_i^{dwell}. Unfortunately, this means that in general, there may be times at which an agent i is forced to remain disconnected even when it does not know how to move to contribute positively to the global task (or it may not even be possible if it is at a local minimum). Remarkably, from the way we have distributed $\dot{V}$ using (7), if agent i sets $u_i(t) = 0$, its instantaneous contribution to the global objective is exactly 0.

Thus, we allow an agent's control to change while it is disconnected and modify the control law described in the previous section as follows. If the chosen ideal communication time $t_i^{\text{ideal}} = (1 - \sigma_i)T_i^{\star} + \sigma_i T_i^{\text{total}}$ is greater than or equal to $t_i^{\ell} + T_i^{\text{dwell}}$, then nothing changes; agent i sets its next communication time $t_i^{\ell+1} = t_i^{\text{ideal}}$ and uses the control law (27) on the entire disconnected interval.

If, on the other hand, $t_i^{\text{ideal}} < t_i^{\ell} + T_i^{\text{dwell}}$, we let agent i use the usual control law until t_i^{ideal}, until which it knows it can make a positive contribution. After t_i^{ideal}, we force agent i to remain still until it has been disconnected for a duration equal to the dwell time. In other words, we set $t^{\text{next}} = t_i^{\ell} + T_i^{\text{dwell}}$, $t^{\text{expire}} = t_i^{\text{ideal}}$ and use control law (27) on the interval $[t_i^{\ell}, t_i^{\text{expire}})$.

For $t \in [t_i^{\text{expire}}, t_i^{\ell+1})$, we then set $u_i(t) = 0$, and note that because $\dot{V}_i(t) = 0$ on this interval, we still have the desired contribution to the global objective

$$\int_{t_i^{\ell}}^{t_i^{\ell+1}} \dot{V}_i(\tau)d\tau = \int_{t_i^{\ell}}^{t_i^{\text{expire}}} \dot{V}_i(\tau)d\tau < 0. \tag{28}$$

Agent i is then still able to calculate the position of any neighbor j exactly for any $t < t_j^{\text{next}}$ using information available on the cloud by

$$x_j(t) = \begin{cases} x_j(t_j^{\text{last}}) + u_j(t_j^{\text{last}})(t - t_j^{\text{last}}) & t < t_j^{\text{expire}} \\ x_j(t_j^{\text{last}}) + u_j(t_j^{\text{last}})(t_j^{\text{expire}} - t_j^{\text{last}}) & \text{otherwise.} \end{cases} \tag{29}$$

An overview of the fully synthesized self-triggered coordination algorithm is presented in Algorithm 1. Next, we present the main convergence result of this algorithm.

Theorem III.4: Given the dynamics (1) and $\mathcal{G}$ connected, if the sequence of times $\{t_i^{\ell}\}$ and control laws $u_i(t_i^{\ell})$ are determined by Algorithm 1 for all $i \in \{1, \ldots, N\}$, then

$$|x_i(t) - x_j(t)| \to 0 \tag{30}$$

for all $i, j \in \{1, \ldots, N\}$ as $t \to \infty$.

Proof: First, because $t^{\ell+1} - t^{\ell} \ge T_i^{\text{dwell}} > 0$, Zeno behavior is impossible, and so $x(t)$ exists for all $t \ge 0$.

Consider ΔV_i^{ℓ} [as defined in (17)], which is the net contribution of agent i over the time interval $[t_i^{\ell}, t_i^{\ell+1})$. From the definition of $T_i^{\star}$ and T_i^{total}, and the continuity of $\dot{V}$, it is clear that we have $T_i^{*} \le t_i^{\text{ideal}} \le T_i^{\text{total}}$. From the definition of T^{total}

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 15:16:35 UTC from IEEE Xplore. Restrictions apply.

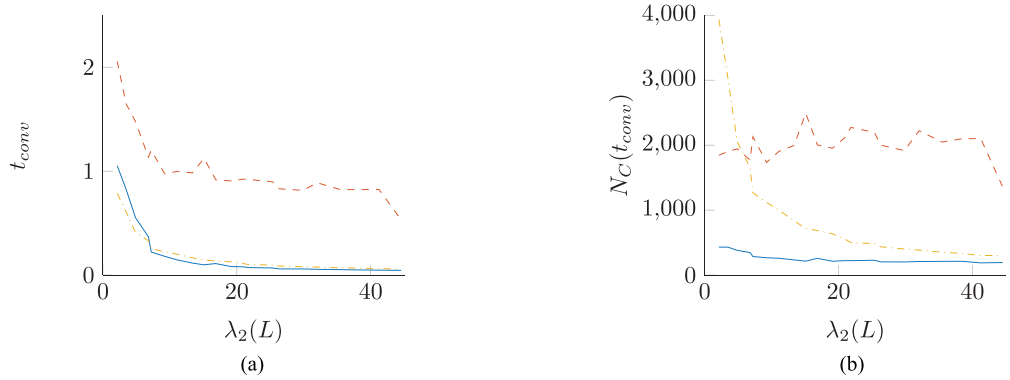


Fig. 1. Plots of (a) the time to convergence, and (b) number of communications required for convergence for our algorithm with $\sigma = 0.5$ and promise function $f(x) = x$ (solid blue), the algorithm presented in [27] with $\sigma = 0.5$ (dashed red), and for a periodic triggering rule with period $T = 0.01$ (dot-dash yellow). The algorithm presented here results in a similar convergence time as a periodic triggering rule, however, with far fewer communications required.

Algorithm 1: III-C.

At $t = 0$, initialize $u_i = 0$, t_i^0 arbitrary small number.
At communication time t_i^ℓ, agent $i \in \{1, \dots, N\}$ performs:

1: download $t_j^{\text{last}}, t_j^{\text{expire}}, t_j^{\text{next}}, x_j(t_j^{\text{last}}), u_i(t_j^{\text{last}}), M_j$ for all $j \in \mathcal{N}_i$ from cloud
2: compute neighbor positions $x_j(t_i^\ell)$ using (29)
3: compute ideal control $u_i^\star(t_i^\ell) = -\sum_{j\in\mathcal{N}_i}(x_i(t_i^\ell) - x_j(t_i^\ell))$
4: compute $u_i^{\max}(t_i^\ell)$ using (26) and saved τ_{ij} data
5: compute control $u_i(t_i^\ell)$ with (27)
6: compute $T_i^\star$ as the solution to (11)
7: compute T^{total} as the solution to (18)
8: set $t_i^{\text{ideal}} = (1-\sigma_i)T_i^\star + \sigma_i T_i^{\text{total}}$
9: **if** $t_i^{\text{ideal}} > t_i^\ell + t_{\max}$ **then**
10: set $t_i^{\text{expire}} = t_i^{\ell+1} = t_i^\ell + t_{\max}$
11: **else if** $t_i^{\text{ideal}} < t_i^\ell + T_i^{\text{dwell}}$ **then**
12: set $t_i^{\text{expire}} = t_i^{\text{ideal}}$
13: set $t_i^{\ell+1} = t_i^\ell + T_i^{\text{dwell}}$
14: **else**
15: set $t_i^{\text{expire}} = t_i^{\ell+1} = t_i^{\text{ideal}}$
16: **end if**
17: upload promise $M_i = |u_i^\star(t_i^\ell)|$ to cloud
18: upload $t_i^{\text{last}} = t_i^\ell$, $t_i^{\text{next}} = t_i^{\ell+1}$, t_i^{expire}, $u_i(t_i^\ell)$, $x_i(t_i^\ell)$ to cloud
19: disconnect and set $u_i(t) = u_i(t_i^\ell)$ for $t \in [t_i^\ell, t_i^{\text{expire}})$, $u_i(t) = 0$ for $t \in [t_i^{\text{expire}}, t_i^{\ell+1})$

and Proposition 2, we have that

$$\int_{t^\ell}^{T_i^{\text{total}}} \dot{V}_i(\tau)d\tau \le 0 \tag{31}$$

and furthermore that

$$\int_{t^\ell}^{t} \dot{V}_i(\tau)d\tau \le 0 \tag{32}$$

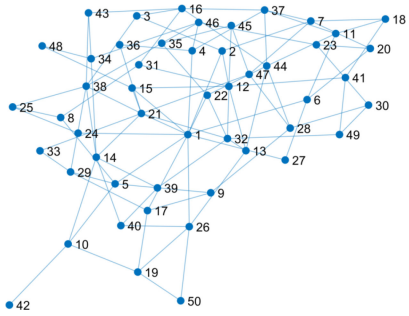


Fig. 2. Simulated communication network.

for any $t \in [t_i^\ell, T_i^{\text{total}})$, and in particular for t_i^{ideal}. Thus, we have

$$\Delta V_i^\ell = \underbrace{\int_{t_i^\ell}^{t_i^{\text{ideal}}} \dot{V}(\tau)d\tau}_{\le 0} + \underbrace{\int_{t_i^{\text{ideal}}}^{t_i^{\ell+1}} \dot{V}(\tau)d\tau}_{=0} \le 0 \tag{33}$$

for all $i \in \{1, \dots, N\}$ and for all $\ell \in \{1, \dots, \ell_i^{\max} - 1\}$.

Now, consider the objective function $V = \frac{1}{2}x^T Lx$. Recall we can decompose it as

$$V(x(t)) = V(x(0)) + \sum_{i=1}^{N} \int_0^t \dot{V}_i(\tau)d\tau. \tag{34}$$

Letting $\ell_i^{\max}(t) = \text{argmax}_{\ell\in\mathbb{Z}_{\ge 0}}\, t_i^\ell \le t$ be the index such that $t_i^{\text{last}}(t) = t_i^{\ell_i^{\max}(t)}$, we can further expand $V(x(t))$ as

$$V(x(t)) = V(x(0)) + \sum_{i=1}^{N} \sum_{\ell=0}^{\ell_i^{\max}(t)} \int_{t_i^\ell}^{\min\{t_i^{\ell+1}, t\}} \dot{V}_i(\tau)d\tau \tag{35}$$

$$= V(x(0)) + \sum_{i=1}^{N} \left[\sum_{\ell=0}^{\ell_i^{\max}-1} \Delta V_i^\ell + \int_{t^{\ell_i^{\max}}}^{t} \dot{V}_i(\tau)d\tau \right]. \tag{36}$$

By the definition of $\ell_i^{\max}$, we must have that $t \le t_i^{\ell_i^{\max}+1}$ for all i. Thus, from (32), the last term in (36) must be nonpositive, and from (33) the second term must be as well.

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 15:16:35 UTC from IEEE Xplore. Restrictions apply.

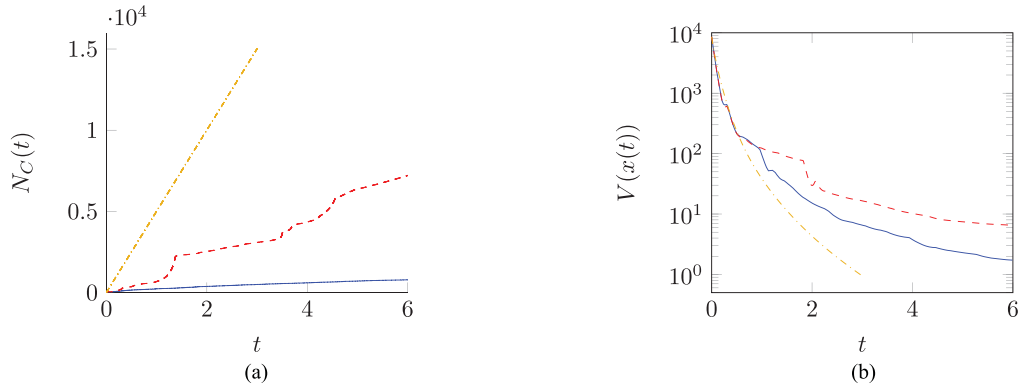


Fig. 3. Plots of (a) the cumulative number of communications, and (b) the full system Lyapunov function for the operation of our algorithm with $\sigma = 0.5$, $f(x) = x$ in solid blue, the algorithm from [27] with $\sigma = 0.5$ in dashed red, and a periodic rule with $T = 0.01$ in dot-dash yellow on the communication graph shown in Fig. 2. Our algorithm exhibits similar convergence speed as the others, however, with far fewer communication events required.

It is now clear that $V(x(t))$ is a nonincreasing function along the system trajectories and bounded below by 0. Furthermore, by the introduction of the dwell time, it is clear that each sequence of times $\{t_i^\ell\}_{\ell \in \mathbb{Z}_{\geq 0}}$ goes to infinity as $\ell \to \infty$. Thus, $\lim_{t\to\infty} V(x(t)) = C \geq 0$ exists.

Because $\Delta V_i^\ell \leq 0$ for all ℓ and $V(x)$ is bounded from below, it is guaranteed that $\Delta V_i^\ell \to 0$ as $t \to \infty$. By LaSalle's Invariance Principle[1] [29], the trajectories of the system converge to the largest invariant set contained in

$$\{x \in \mathbb{R}^N \mid \dot{V}_i(x) = 0 \; \forall i \in \{1, \ldots, N\}\}. \tag{37}$$

From examination of the local objective contribution (7), we see that $\dot{V}_i(x) = 0$ if and only if either $u_i(t) = 0$ or $\sum_{j\in\mathcal{N}_i} x_j - x_i = 0$. First, note that $\sum_{j\in\mathcal{N}_i} x_j - x_i = 0$ for all i if and only if the system is at consensus. This condition is equivalent to $Lx = 0$, and as we assume $\mathcal{G}$ is connected, $\ker(L) = \{\mathbf{1}_N\}$.

Now, assume the system is not at consensus, so there is at least one agent i with $\sum_{j\in\mathcal{N}_i} x_j - x_i \neq 0$. From the control law (3), it is clear that this implies the next time agent i communicates t_i^ℓ, we will have $u_i^\star(t_i^\ell) \neq 0$ as well. Thus, we simply have to prove that the next time t_i^ℓ that agent i communicates, it computes a $t_i^{\text{ideal}} > t_i^\ell$ so the real control $u_i(t) \neq 0$ for a nonzero period of time. As $t_i^{\text{ideal}} \geq T_i^\star$, the last time at which we can guarantee $\dot{V}_i(t) \leq 0$, it suffices to show $T_i^\star > t_i^\ell$.

$T_i^\star$ is computed as the earliest time after which our bound on $\dot{V}_i(t)$ is positive. From (43), we can write this bound at time t_i^ℓ as $\dot{V}_i(t_i^\ell) = -(\sum_{j\in\mathcal{N}_i} x_j - x_i)^2$, which is strictly negative. As the bound is a continuous function of time, this implies that the smallest t for which we can no longer guarantee $\dot{V}_i(t) \leq 0$ is strictly greater than t_i^ℓ. Thus, the next time agent i communicates, it will apply a nonzero control for a positive duration.

Finally, due to the existence of the maximum disconnected time $t_{\max}$, we know that there exists a finite future time at which agent i will communicate. ■

[1]Due to the discrete communication events, the evolution of V is not continuously differentiable and the aforementioned theorem does not strictly apply. The more formal proof is obtained under the theory of hybrid systems; however, the idea and results are the same, and it is not done here for space and clarity.

IV. Simulation

In this section, we simulate a system of 50 agents with various communication graph topologies, and with all $\sigma_i = \sigma = 0.5$. In all simulations, we set $T^{\text{dwell}} = 10^{-8}$ s, but the dwell time condition was never used. Similarly, we set $t_{\max} = 5$ s, but it had no effect on any of the simulations. Beginning with the same random initial state x_i for all agents and a minimally connected tree communication graph, we progressively generated increasingly connected graphs by randomly adding edges to the graph until we reached the complete graph K_N. We simulated our algorithm on multiple communication topologies generated in this way, using the second-smallest eigenvalue of the graph Laplacian $\lambda_2(L)$ as a metric for the graph's connectivity. We additionally compare our algorithm with the algorithm proposed in [27] (also with $\sigma = 0.5$), as well as with a simple periodic triggering rule where each agent communicates every $T = 0.01$ s and uses the constant control law $u_i^\star(t_i^\ell)$ on each interval.

We define a heuristic convergence criterion as when the mean of all distances from the mean state is less than some threshold ε, i.e., $\frac{1}{N}\sum_i \|x_i - \bar{x}\| < \varepsilon$; in the simulations here, we set $\varepsilon = 0.2$ m. Let t_{conv} be the first time at which the above-mentioned convergence criterion is satisfied, and let $N_C(t_{\text{conv}})$ be the total number of communications by all agents that happened up to time t_{conv}. The convergence times t_{conv} can be seen in Fig. 1(a), and the total number of communications required for convergence is shown in Fig. 1(b). We see that for all communication graph topologies, our algorithm clearly outperforms the algorithm presented in [27]. Furthermore, though with a period of $T = 0.01$ our algorithm converges in a similar time as the simple periodic algorithm, it does so with far fewer communication events required and without requiring global knowledge of the communication graph (the periodic algorithm is only guaranteed to converge for $T < 2/\lambda_{\max}(L)$ [9]).

To view the operation of our algorithm on a single system, we simulate a system of 50 agents on a relatively sparsely connected communication graph ($\lambda_2(L) = 0.53$, average node degree is 3.9) seen in Fig. 2. We ran our algorithm, the algorithm presented in [27], and a periodic triggering algorithm on this

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 15:16:35 UTC from IEEE Xplore. Restrictions apply.

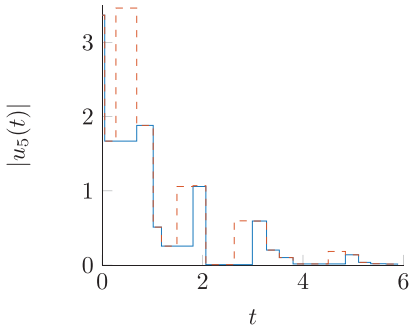


Fig. 4. Magnitude of control law in use by agent $i = 5$, $|u_5(t)|$ (solid blue), as well as its current promise on the cloud server $M_5(t)$ (dashed red).

system with the same parameters as in the previous paragraphs. The number of cumulative communications $N_C(t)$ is shown in Fig. 3(a), and the evolution of the Lyapunov function $V(x(t))$ is seen in Fig. 3(b). As before, our algorithm results in convergence speed comparable to the periodic triggering rule with a short communication period, however, with far fewer communications required and no global information needed. Note that although all given graphs are from just one run of the simulation, the trends shown hold for other complex random communication graphs and initial states.

Finally, to demonstrate the effect of promises on the control inputs, a single agent's control law (agent 5) from a run of our algorithm with $\sigma = 0.5$ is shown in Fig. 4, along with its "promise" currently on the cloud server. There exists a lag between when the promised control max $M_5(t)$ increases and when the actual control increases likewise. While $M_5(t)$ represents the ideal control that the agent would use, it is still bound to a previous promise until the newer one propagates to all neighbor agents.

V. Conclusion

We have presented a novel self-triggering algorithm that, given only the ability to communicate asynchronously at discrete intervals through a cloud server, provably drives a set of agents to consensus without Zeno behavior. Unlike most previous work, we do not require an agent to be able to listen continuously, instead only being able to receive information at its discrete communication times. Through the use of control promises, we are able to bound the states of neighboring agents, allowing an agent to remain disconnected until its total contribution to the consensus would become detrimental. The given algorithm requires no global parameters and is fully distributed, requiring no computation to be done off of each local platform. Simulation results show the effectiveness of the proposed algorithm. In the future, we are interested in investigating control laws different from (27) and forms of $f(x)$ other than $f(x) = cx$ that may be able to provide more infrequent communication or faster convergence and in guaranteeing no Zeno behavior without a dwell time.

Appendix A
Proof of Proposition 1

We begin by further splitting up the local objective contribution $\dot{V}_i$ as a sum of individual neighbor pair contributions: $\dot{V}_i(t) = \sum_{j\in\mathcal{N}_i} \dot{V}_{ij}(t)$, where

$$\dot{V}_{ij}(t) \triangleq -u_i(t)\left(x_j(t) - x_i(t)\right). \tag{38}$$

For $t \le t_j^{\text{next}}$, we can write $\dot{V}_{ij}(t)$ exactly as

$$\begin{aligned}\dot{V}_{ij}(t) = &-u_i(t)\left[x_j(t_i^{\text{last}}) - x_i(t_i^{\text{last}})\right.\\ &\left.+(u_j(t) - u_i(t))(t - t_i^{\text{last}})\right].\end{aligned} \tag{39}$$

For $t > t_j^{\text{next}}$, since agent i no longer has access to $u_j(t)$, we write it as follows:

$$\begin{aligned}\dot{V}_{ij}(t) = &-u_i(t)\Big[x_j(t_j^{\text{next}}) + \int_{t_j^{\text{next}}}^{t} u_j(\tau)d\tau\\ &-\left(x_i(t_j^{\text{next}}) + u_i(t)(t - t_j^{\text{next}})\right)\Big].\end{aligned} \tag{40}$$

We can then use the promise $M_j(t)$ to bound

$$\left|\int_{t_j^{\text{next}}}^{t} u_j(\tau)d\tau\right| \le M_j(t)(t - t_j^{\text{next}}) \tag{41}$$

allowing us to upper bound $\dot{V}_{ij}(t)$ for $t > t_j^{\text{next}}$ with

$$\begin{aligned}\dot{V}_{ij}(t) \le &-u_i(t)(x_j(t_j^{\text{next}}) - x_i(t_j^{\text{next}}))\\ &+(|u_i(t)|M_j(t) + u_i(t)^2)(t - t_j^{\text{next}}).\end{aligned} \tag{42}$$

Letting α_{ij}, β_{ij}, and γ_{ij} be as defined in Proposition 1, we can write these as

$$\dot{V}_{ij}(t) \le \begin{cases}\alpha_{ij}(t_i^{\text{last}}) + \beta_{ij}(t_i^{\text{last}})(t - t_i^{\text{last}}) & t \le t_j^{\text{next}}\\ \alpha_{ij}(t_j^{\text{next}}) + \gamma_{ij}(t_i^{\text{last}})(t - t_j^{\text{next}}) & \text{otherwise.}\end{cases} \tag{43}$$

Assume that the solution to (11) lies in the interval $T_i^* \in [t_{\pi(k)}^{\text{next}}, t_{\pi(k+1)}^{\text{next}})$ for some $k \in \{0, \ldots, |\mathcal{N}_i|\}$. On this interval, the states of neighbors $\pi(m)$ for $m > k$ are known exactly, while those $\pi(m')$ with $m' \le k$ are only known to lie in a ball since they are scheduled to communicate and change their control by this time interval. Using (39) and (42), we can write the local objective contribution $\dot{V}_i(t)$ for t in this interval as

$$\dot{V}_i(t) = \sum_{m'=1}^{k} \dot{V}_{i\pi(m')}(t) + \sum_{m=k+1}^{|\mathcal{N}_i|} \dot{V}_{i\pi(m)}(t) \tag{44}$$

$$\begin{aligned}&\le \sum_{m'=1}^{k}\left(\alpha_{i\pi(m')}\left(t_{\pi(m')}^{\text{next}}\right) + \gamma_{i\pi(m')}\left(t_i^{\text{last}}\right)\left(t - t_{\pi(m')}^{\text{next}}\right)\right)\\ &\quad+\sum_{m=k+1}^{|\mathcal{N}_i|}\left(\alpha_{i\pi(m)}\left(t_i^{\text{last}}\right) + \beta_{i\pi(m)}\left(t_i^{\text{last}}\right)\left(t - t_i^{\text{last}}\right)\right).\end{aligned} \tag{45}$$

Let $\tau_i^{\star,(k)}$ be the solution to (11) with the additional constraint that t be within the interval $[t_{\pi(k)}^{\text{next}}, t_{\pi(k+1)}^{\text{next}})$. The objective derivative constraint in (11) can be rewritten using (45) in

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 15:16:35 UTC from IEEE Xplore. Restrictions apply.

the form seen in Proposition 1. The solution $T_i^\star$ to the original optimization (11) can then be written as

$$T_i^\star = \min\{\tau_i^{\star,(k)} \mid k \in \{0,\dots,|\mathcal{N}_i|\}\}. \tag{46}$$

Appendix B
Proof of Proposition 2

First, note that the separation amongst neighbor pairs is still valid

$$\int_{t_i^{\text{last}}}^{t} \dot{V}_i(\tau)d\tau = \sum_{j\in\mathcal{N}_i}\int_{t_i^{\text{last}}}^{t} \dot{V}_{ij}(\tau)d\tau. \tag{47}$$

For an agent $j \in \mathcal{N}_i$ and $t \le t_j^{\text{next}}$, we can exactly compute the pair contribution over its disconnected interval as

$$\begin{aligned}\int_{t_i^{\text{last}}}^{t} \dot{V}_{ij}(\tau)d\tau &= \alpha_{ij}(t_i^{\text{last}})(t-t_i^{\text{last}})\\ &\quad+\frac{1}{2}\beta_{ij}(t_i^{\text{last}})(t-t_i^{\text{last}})^2\end{aligned} \tag{48}$$

where α_{ij} and β_{ij} are as given in (13) and (14).

For agent j and $t > t_j^{\text{next}}$, we first split the integral into a part that we can compute exactly and a part that we can only bound as

$$\int_{t_i^{\text{last}}}^{t} \dot{V}_{ij}(\tau)d\tau = \int_{t_i^{\text{last}}}^{t_j^{\text{next}}} \dot{V}_{ij}(\tau)d\tau + \int_{t_j^{\text{next}}}^{t} \dot{V}_{ij}(\tau)d\tau \tag{49}$$

and using the bound (45) can write

$$\begin{aligned}\int_{t_j^{\text{next}}}^{t} \dot{V}_{ij}(\tau)d\tau &\le \alpha_{ij}(t_j^{\text{next}})(t-t_j^{\text{next}})\\ &\quad+\frac{1}{2}\gamma_{ij}(t_i^{\text{last}})(t-t_j^{\text{next}})^2.\end{aligned} \tag{50}$$

The total pair contribution $\int \dot{V}_{ij}(t)$ is then given by (48) for $t \le t_j^{\text{next}}$ and bounded by

$$\begin{aligned}\int_{t_i^{\text{last}}}^{t} \dot{V}_{ij}(\tau)d\tau &\le \alpha_{ij}(t_i^{\text{last}})(t_j^{\text{next}}-t_i^{\text{last}})\\ &\quad+\frac{1}{2}\beta_{ij}(t_i^{\text{last}})(t_j^{\text{next}}-t_i^{\text{last}})^2+\alpha_{ij}(t_j^{\text{next}})(t-t_j^{\text{next}})\\ &\quad+\frac{1}{2}\gamma_{ij}(t_i^{\text{last}})(t-t_j^{\text{next}})^2\end{aligned} \tag{51}$$

for $t > t_j^{\text{next}}$.

As before, consider times t that lie in the interval $[t_{\pi(k)}^{\text{next}}, t_{\pi(k+1)}^{\text{next}})$ for some $k \in \{0,\dots,|\mathcal{N}_i|\}$. We can bound the full objective contribution for times t in this interval as [from (48), (49), and (51)]

$$\begin{aligned}\int_{t_i^{\text{last}}}^{t} \dot{V}_i(\tau)d\tau \le \sum_{m'=1}^{k}&\Big[\alpha_{i\pi(m')}(t_i^{\text{last}})(t_{\pi(m')}^{\text{next}}-t_i^{\text{last}})\\ &+\frac{1}{2}\beta_{i\pi(m')}(t_i^{\text{last}})(t_{\pi(m')}^{\text{next}}-t_i^{\text{last}})^2\\ &+\alpha_{i\pi(m')}(t_{\pi(m')}^{\text{next}})(t-t_{\pi(m')}^{\text{next}})\\ &+\frac{1}{2}\gamma_{i\pi(m')}(t_i^{\text{last}})(t-t_{\pi(m')}^{\text{next}})^2\Big]\\ &+\sum_{m=k+1}^{|\mathcal{N}_i|}\Big[\alpha_{i\pi(m)}(t_i^{\text{last}})(t-t_i^{\text{last}})\\ &+\frac{1}{2}\beta_{i\pi(m)}(t_i^{\text{last}})(t-t_i^{\text{last}})^2\Big].\end{aligned} \tag{52}$$

Let $\tau_i^{tot,(k)}$ be the optimal solution to (18) with the additional constraint that t be within the interval $[t_{\pi(k)}^{\text{next}}, t_{\pi(k+1)}^{\text{next}})$. Using (52), we can rewrite the objective bound in (18) on this interval, resulting in the optimization seen in (19). The solution T_i^{total} to (18) can then be written as

$$T_i^{\text{total}} = \min\{\tau_i^{tot,(k)} \mid k \in \{0,\dots,|\mathcal{N}_i|\}\}. \tag{53}$$

References

[1] C. Nowzari, E. Garcia, and J. Cortés, "Event-triggered communication and control of networked systems for multi-agent consensus," *Automatica*, vol. 105, pp. 1–27, 2019.

[2] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.

[3] M. Mesbahi and M. Egerstedt, *Graph Theoretic Methods in Multiagent Networks* (Series Applied Mathematics Series). Princeton, NJ, USA: Princeton Univ. Press, 2010.

[4] R. Olfati-Saber and R. M. Murray, "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1520–1533, Sep. 2004.

[5] K. J. Åström and B. M. Bernhardsson, "Comparison of Riemann and Lebesgue sampling for first order stochastic systems," in *Proc. IEEE Conf. Decis. Control*, Las Vegas, NV, USA, Dec. 2002, pp. 2011–2016.

[6] W. P. M. H. Heemels, K. H. Johansson, and P. Tabuada, "An introduction to event-triggered and self-triggered control," in *Proc. IEEE Conf. Decis. Control*, Maui, HI, USA, 2012, pp. 3270–3285.

[7] M. Mazo Jr. and P. Tabuada, "Decentralized event-triggered control over wireless sensor/actuator networks," *IEEE Trans. Autom. Control*, vol. 56, no. 10, pp. 2456–2461, Oct. 2011.

[8] X. Wang and M. D. Lemmon, "Event-triggering in distributed networked control systems," *IEEE Trans. Autom. Control*, vol. 56, no. 3, pp. 586–601, Mar. 2011.

[9] G. Xie, H. Liu, L. Wang, and Y. Jia, "Consensus in networked multi-agent systems via sampled control: Fixed topology case," in *Proc. Amer. Control Conf.*, St. Louis, MO, USA, 2009, pp. 3902–3907.

[10] W. P. M. H. Heemels and M. C. F. Donkers, "Model-based periodic event-triggered control for linear systems," *Automatica*, vol. 49, no. 3, pp. 698–711, 2013.

[11] X. Meng and T. Chen, "Event based agreement protocols for multi-agent networks," *Automatica*, vol. 49, no. 7, pp. 2125–2132, 2013.

[12] M. Zhong and C. G. Cassandras, "Asynchronous distributed optimization with event-driven communication," *IEEE Trans. Autom. Control*, vol. 55, no. 12, pp. 2735–2750, Dec. 2010.

[13] A. Anta and P. Tabuada, "To sample or not to sample: Self-triggered control for nonlinear systems," *IEEE Trans. Autom. Control*, vol. 55, no. 9, pp. 2030–2042, Sep. 2010.

[14] X. Wang and M. D. Lemmon, "Self-triggered feedback control systems with finite-gain L_2 stability," *IEEE Trans. Autom. Control*, vol. 54, no. 3, pp. 452–467, Mar. 2009.

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 15:16:35 UTC from IEEE Xplore. Restrictions apply.

[15] C. Nowzari and J. Cortés, "Self-triggered coordination of robotic networks for optimal deployment," *Automatica*, vol. 48, no. 6, pp. 1077–1087, 2012.
[16] D. V. Dimarogonas, E. Frazzoli, and K. H. Johansson, "Distributed event-triggered control for multi-agent systems," *IEEE Trans. Autom. Control*, vol. 57, no. 5, pp. 1291–1297, May 2012.
[17] G. S. Seybotha, D. V. Dimarogonas, and K. H. Johansson, "Event-based broadcasting for multi-agent average consensus," *Automatica*, vol. 49, no. 1, pp. 245–252, 2013.
[18] E. Garcia, Y. Cao, H. Yu, P. Antsaklis, and D. Casbeer, "Decentralised event-triggered cooperative control with limited communication," *Int. J. Control*, vol. 86, no. 9, pp. 1479–1488, 2013.
[19] C. Nowzari and J. Cortés, "Zeno-free, distributed event-triggered communication and control for multi-agent average consensus," in *Proc. Amer. Control Conf.*, Portland, OR, USA, 2014, pp. 2148–2153.
[20] J. Peters, S. J. Wang, A. Surana, and F. Bullo, "Cloud-supported coverage control for persistent surveillance missions," *J. Dyn. Syst., Meas., Control*, vol. 139, pp. 1–12, Feb. 2017.
[21] P. V. Teixeira, D. V. Dimarogonas, K. H. Johansson, and J. Sousa, "Event-based motion coordination of multiple underwater vehicles under disturbances," in *Proc. IEEE OCEANS*, Sydney, NSW, Australia, 2010, pp. 1–6.
[22] A. Adaldo, D. Liuzza, D. V. Dimarogonas, and K. H. Johansson, "Control of multi-agent systems with event-triggered cloud access," in *Proc. Eur. Control Conf.*, Linz, Austria, 2015, pp. 954–961.
[23] C. Nowzari and G. J. Pappas, "Multi-agent coordination with asynchronous cloud access," in *Proc. Amer. Control Conf.*, Jul. 2016, pp. 4649–4654.
[24] A. Adaldo, D. Liuzza, D. V. Dimarogonas, and K. H. Johansson, "Multi-agent trajectory tracking with self-triggered cloud access," in *Proc. IEEE 55th Conf. Decis. Control*, Dec. 2016, pp. 2207–2214.
[25] A. Adaldo, D. Liuzza, D. V. Dimarogonas, and K. H. Johansson, "Cloud-supported formation control of second-order multi-agent systems," *IEEE Trans. Control Netw. Syst.*, vol. 5, no. 4, pp. 1563–1574, Dec. 2018.
[26] C. Nowzari and J. Cortés, "Team-triggered coordination for real-time control of networked cyberphysical systems," *IEEE Trans. Autom. Control*, vol. 61, no. 1, pp. 34–47, Jan. 2016.
[27] S. L. Bowman, C. Nowzari, and G. J. Pappas, "Coordination of multi-agent systems via asynchronous cloud communication," in *Proc. IEEE 55th Conf. Decis. Control*, Dec. 2016, pp. 2215–2220.
[28] X. Meng, L. Xie, Y. C. Soh, C. Nowzari, and G. J. Pappas, "Periodic event-triggered average consensus over directed graphs," in *Proc. IEEE Conf. Decis. Control*, Osaka, Japan, Dec. 2015, pp. 4151–4156.
[29] H. Khalil, *Nonlinear Systems* (Series Pearson Education). Englewood Cliffs, NJ, USA: Prentice-Hall, 2002. [Online]. Available: https://books.google.com/books?id=t_d1QgAACAAJ



Sean L. Bowman (S'12) received the B.Comp.E. degree in computer engineering from the University of Minnesota, Minneapolis, MN, USA. He is currently working toward the Ph.D. degree in the Computer and Information Science Department, the University of Pennsylvania, Philadelphia, PA, USA.

His current research interests include distributed coordination algorithms, vision-aided inertial navigation, and localization and mapping over semantic maps.

Mr. Bowman was the recipient of the Best Conference Paper Award at the 2017 IEEE International Conference on Robotics and Automation.



Cameron Nowzari (M'10) received the Ph.D. degree in mechanical engineering from the University of California, San Diego, San Diego, CA, USA, in September 2013.

He then held a Postdoctoral position with the Electrical and Systems Engineering Department, University of Pennsylvania, until 2016. He is currently an Assistant Professor with the Electrical and Computer Engineering Department, George Mason University, Fairfax, VA, USA. His current research interests include dynamical systems and control, distributed coordination algorithms, robotics, event- and self-triggered control, Markov processes, network science, spreading processes on networks, and the Internet of Things.

Dr. Nowzari has received several awards including the American Automatic Control Council's O. Hugo Schuck Best Paper Award and the *IEEE Control Systems Magazine* Outstanding Paper Award.



George J. Pappas (F'09) is the Joseph Moore Professor and Chair of the Department of Electrical and Systems Engineering, University of Pennsylvania, Philadelphia, PA, USA. He also holds a secondary appointment with the Departments of Computer and Information Sciences, and Mechanical Engineering and Applied Mechanics. He is member of the GRASP Lab and the PRECISE Center. He previously served as the Deputy Dean for Research with the School of Engineering and Applied Science. His research focuses on control theory and, in particular, hybrid systems, embedded systems, hierarchical and distributed control systems, with applications to unmanned aerial vehicles, distributed robotics, green buildings, and biomolecular networks.

Mr. Pappas was the recipient of various awards such as the Antonio Ruberti Young Researcher Prize, the George S. Axelby Award, the O. Hugo Schuck Best Paper Award, the National Science Foundation PECASE, and the George H. Heilmeier Faculty Excellence Award.

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 15:16:35 UTC from IEEE Xplore. Restrictions apply.