IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received November 14, 2021, accepted December 12, 2021, date of publication December 27, 2021,
date of current version February 24, 2022.

Digital Object Identifier 10.1109/ACCESS.2021.3138857

Communication-Aware Consensus-Based Decentralized Task Allocation in Communication Constrained Environments

SHARAN RAJA, (Member, IEEE), GOLNAZ HABIBI, (Member, IEEE), AND JONATHAN P. HOW, (Fellow, IEEE)
Aerospace Controls Laboratory, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

Corresponding author: Sharan Raja (sharanraja@alum.mit.edu)

This work was supported by Lockheed Martin, ARL Distributed and Collaborative Intelligent Systems (DCIST) under Cooperative Agreement Number W911NF-17-2-0181, and computational support through Amazon Web Services.

ABSTRACT Most of the consensus-based task allocation algorithms assume reliable and unlimited communication between the agents. However, this assumption can be easily violated in real environment with limited bandwidth and message collisions. This paper presents a deep reinforcement learning framework in which agents learn how to schedule and censor themselves amongst the other agents competing for access to a limited communication medium. In particular, the process learns to schedule the communication between agents to improve the performance of task allocation in environments with constrained communication in terms of limited bandwidth and message collision. The proposed approach, called Communication-Aware Consensus-Based Bundle Algorithm (CA-CBBA), extends the previous CBBA that the learned communication policy enables efficient utilization of the shared medium by prioritizing agents with messages that are important for the mission. Furthermore, agents in denser parts of the network are censored appropriately to alleviate the message collision and hidden node problems. We evaluate our approach in various task assignment scenarios, and the results show that CA-CBBA outperforms CBBA in terms of convergence time, rate of conflict resolution, and task allocation reward. Moreover, we show that CA-CBBA yields a policy that generalizes beyond the training set to handle larger team sizes. Finally, the results on time-critical problems, such as a search-and-rescue mission, show that CA-CBBA also outperforms the baselines considered (e.g., CBBA, MCDGA, and ACBBA) in terms of number of unassigned and conflicted tasks in most of the scenarios.

INDEX TERMS Decentralized task allocation, constrained communication, multi-agent reinforcement learning, censoring message, value of information.

I. INTRODUCTION

Multi-Robot Task Allocation (MRTA) has been used in a wide range of applications such as defense [4], search-and-rescue [5], [6], agricultural spraying [7], and surveillance [8]. MRTA can be viewed as a generalized version of the Traveling Salesman Problem or Vehicle Routing Problem, both of which are shown to be NP-Hard [9], [10]. Hence, finding optimal solution to MRTA becomes computationally intractable. The state of the art algorithms trade-off optimality for reduced algorithmic complexity. Several centralized approaches that use particle swarm optimization [11], [12] and genetic algorithms [13]–[15] have been developed to solve MRTA. However, they require the agents continuously communicate to a central server that solves the planning problem and then sends the instructions back. This central planner makes the algorithm not be scalable to larger teams as agents have to stay within its communication range. Furthermore, centralized approaches are susceptible to single point failures. For instance, an adversary can derail the planning mission by successfully attacking only the central planner.

Decentralized algorithms based on market-based approaches [16]–[18], bio-nspired approaches [19], [20], and consensus algorithms [21] have been proposed to address the issues associated with centralized approaches. However, most of the decentralized approaches assume the communication is reliable with unlimited bandwidth, or they assume the agents are communicating with the agents in a fully connected

The associate editor coordinating the review of this manuscript and approving it for publication was Gang Mei.

network during the task allocation [22]. Such assumptions can be easily violated in real world scenarios such as search and rescue mission when the task allocation can be operated among agents with limited range of communication or communication channel with limited bandwidth. Ref. [23]–[25] showed the detrimental effects of communication constraints posed by realistic communication environments on the performance of task allocation algorithms. Ref. [26] analyzed the effect of lossy communication between the auctioneer and bidders on solution quality in auction based task allocation problem and showed that the quality of different auctions degrade in different ways. Ref. [23] simulated communication in the ns-3 network simulator and showed that performance degrades as the team size increases due to increased message collisions and channel errors. In this work, we address two communication constraints in networked agents;(i) communication bandwidth and (ii) message collision, by presenting a new learning-based algorithm, *Communication-aware* consensus based bundle algorithm (CA-CBBA), for decentralized task allocation. The main contributions of this paper are:

1) By formulating the communication policy as a multi-agent deep reinforcement learning, we co-design CBBA and the communication policy used by the agents to improve the performance of task allocation in realistic communication networks.
2) Leverage two local features, namely *local communication graph density* (later called the Bron-Kerbosch feature) and *Value of Message* (VoM) to learn a decentralized communication policy that adaptively allocates communication resources across the team to achieve efficient message passing.
3) CA-CBBA prioritizes agents with useful information and censors agents that can cause the communication medium to be clogged up, improving the throughput by $\approx$15% and the convergence time by $\approx$10x compared to baseline CBBA in ad-hoc networks.
4) Demonstrate that CA-CBBA outperforms other baselines in a time-sensitive application such as search and rescue and time scheduling problems in most of the scenarios.
5) The policy learned from CA-CBBA can be generalized to different team sizes beyond the size of the training set and different task numbers.

A. OUTLINE

The article is organized as follows. Related work is presented in Section II. In section III, we review related background such as CBBA and communication protocols that are relevant to understand following sections. In Section IV, we propose learning-based approach, CA-CBBA, which uses two features of local communication graph density and value of message, to learn an efficient communication policy for agents running CBBA under communication constraints. Section V presents a comparison of convergence properties of CA-CBBA against other baselines. This is followed by an ablation study that identifies the contribution of each component of the algorithm. Additionally, we present applications of CA-CBBA to search and rescue and time scheduling problems. Concluding remarks are provided in Section VI.

II. RELATED WORK

Choi *et al.* [1] proposed CBBA by combining market-based and consensus-based algorithms. With assumption on perfect communication, it is proven CBBA achieves feasible solutions identical to the centralized greedy algorithm. Under the diminishing marginal gain (DMG) [27] assumption on marginal scoring function, CBBA is proven to converge to a conflict-free assignment and be robust to both the inconsistencies in situation awareness across the fleet and variations in the communication network topology.

CBBA has been extended to other complex, uncertain and dynamic environments. Bertuccelli *et al.* [28] presented techniques to avoid polygonal obstacles during mission using CBBA. To deal with asynchronous communications, Johnson *et al.* [3], [29] developed rules for conflict resolution in asynchronous networks. The proposed local de-confliction rules introduce rebroadcasting and not broadcasting options to deal with ambiguous timestamps and prevent unnecessary communication. Furthermore, this method can obtain sub-optimal solutions in polynomial time, making it well suited to real-time dynamic environments. Buckman *et al.* [30] introduced CBBA with partial replanning (CBBA-PR) to include dynamic tasks that appear after or while the team is in the process of allocating previously known tasks. CBBA-PR reallocates a portion of previous allocation before each iteration to trade off the solution quality and algorithm convergence in dynamic environments. An extension of CBBA, termed heterogeneous robots consensus-based allocation (HRCA) [31] handles heterogeneous robot networks. Unlike CBBA, the bundle construction phase and conflict resolution phase of HRCA disregard the constraint on the maximum number of tasks assigned to each agent and a bundle resize phase is performed after to handle the associate constraint violations.

This work extends CBBA to execute in environment with communication constraints including the bandwidth and message collision. In our approach, each agent schedules and censors itself based on the network density and its message importance, which shows the improvement of the CBBA in terms of number of conflicted tasks and convergence time.

There exist several work that addressed different constraints in task allocation such as resource constraints [32], [33], time and spatial constraints [34]–[36], and envy minimization [37]. Previous attempts at addressing communication constraints in multiagent algorithms can be classified into two categories. The first is what we classify as the *Algorithm approach* [38]–[43] that attempts to reduce the number of messages by selectively allowing informative agents or censoring uninformative agents.

Kim *et al.* [2] presented MCDGA that modifies the consensus phase of the original CBBA algorithm to prune unwanted messages. Agents in MCDGA unicast (to a specific agent) or broadcast (to neighbors) their messages based on the local communication rules defined by the algorithm. Recently, [22] used meta reasoning approach to switch between different task allocation algorithms based on the communication quality, but that work assumed the agents have a fully connected network. Our work relaxes that assumption and can be utilized in ad-hoc networks that might suffer from message collision and the hidden node problem.

Ref. [44] presented grouped CBBA (G-CBBA) for grouping the UAVs based on their task preference achieved by the initial guesses. The results shows the amount of communication reduced compared to CBBA. However, this approach requires information about UAV's task preference and it cannot be useful for homogeneous agents with no preference in accomplishing the tasks. In general, the algorithm approaches do not explicitly model the communication protocol and suffer from issues such as latency and low network throughput.

An alternative direction is called the *Communication protocol approach*, which attempts to model communication protocols and find optimal protocol parameters that work best for the given multiagent algorithm [45]–[47]. For example, [23] compared the performance of CBBA when using different transport layer protocols and concluded that TCP with IEEE 802.11b unicast reduced message collisions. Although, these approaches aim at incorporating communication constraints, they do not utilize information about the algorithm to filter uninformative messages resulting in a ''clogged up'' communication medium. Although several approaches in communication protocols consider handshaking mechanism such as RTS/CTS in CSMA/CA protocol to reduce the hidden node problem [48], this handshaking introduces latency in message passing, which is a limitation of these protocols in real-time and time-critical task allocations, such as search and rescue.

In recent years, deep learning-based methods have been used for learning the communications in environments with limited bandwidth or limited resources [49]–[51]. The work by Foerster *et al.* [52] is considered as the first attempt that used deep reinforcement learning (DRL) for learning communication protocols. Kim *et al.* [39] presented a learning-based method for scheduling the communication for RL frameworks based on value of importance. Our work instead learns scheduling and censoring to improve the quality of task allocation algorithms, *i.e.* CBBA.

Our work, to the best of our knowledge, is the first DRL method to combine communication protocol and algorithm protocol approaches to improve decentralized task allocation performance, when agents are communicated via Wireless ad-hoc communication. Although our model is customized for improving the CBBA reward function, but it can be extended to other task allocation approaches with different objective function, which is left for future work.

III. BACKGROUND

A. DECENTRALIZED TASK ALLOCATION

The goal of task allocation is to find a conflict-free matching for a set of N_t tasks ($\mathcal{J}$) to a set of N_u agents ($\mathcal{I}$) that maximizes some global reward. Each agent can take up to L_t tasks defined by physical limitation or planning horizon. If $x_{ij} \in \{0, 1\}$ is the decision variable that indicates whether task j is assigned to agent i and c_{ij} is the reward for assigning task j to agent i, the problem can be stated as

$$\begin{aligned} \max & \sum_{i=1}^{N_u} \left(\sum_{j=1}^{N_t} c_{ij}(\boldsymbol{x}_{ij}, \boldsymbol{\rho}_i) x_{ij} \right) \\ \text{s.t,} & \sum_{j=1}^{N_t} x_{ij} \leq L_t, \quad \forall i \in \mathcal{I} \\ & \sum_{i=1}^{N_u} x_{ij} \leq 1, \quad \forall j \in \mathcal{J} \\ & \sum_{i=1}^{N_u} \sum_{j=1}^{N_t} x_{ij} = \min\{N_u L_t, N_t\} = N_{min} \\ & x_{ij} \in \{0, 1\}, \quad \forall (i, j) \in \mathcal{I} \times \mathcal{J} \end{aligned} \tag{1}$$

$\boldsymbol{\rho}_i$ is the list of tasks allocated to agent i in order of their execution. An assignment is conflict-free when each task is assigned to no more than one agent (second constraint in Eq.1. CBBA [1] uses an auction-based task selection and consensus-based conflict resolution process to solve the decentralized version of the above problem.

CBBA consists of two phases - *the bundle construction phase* that utilizes greedy task selection to build task sequence for each robot and *the conflict resolution phase* that uses a consensus routine based on local communication to achieve global agreement.

In every round of CBBA, each agent i shares its state information (winning bids list $\boldsymbol{y}_i \in \mathbb{R}^{N_t}$, winning agent list $\boldsymbol{z}_i \in \mathcal{I}^{N_t}$ and time-stamp of the recent information exchange $\boldsymbol{s}_i \in \mathbb{R}^{N_u}$) with its neighbors. In the conflict resolution phase of CBBA, these messages are used to resolve conflicts based on an extensive set of rules listed in [1]. If D is the diameter of the communication graph formed by the agents, CBBA is shown to converge to the same solution as centralized greedy algorithm SGA in $N_{min}D$ rounds in the worst-case scenario. However, it is assumed there is no constraints in communication between agents.

B. COMMUNICATION CONSTRAINTS

The main contribution of this work is to co-design the CSMA/CA protocol to address two challenges in communication constrained environments. In our model, we assume agents running CSMA/CA medium access control scheme, and UDP is considered as the transport layer protocol. Although TCP provides reliable data transmission, we use UDP as it allows for broadcasting which is inherently efficient for consensus algorithms. Furthermore, [23] shows that

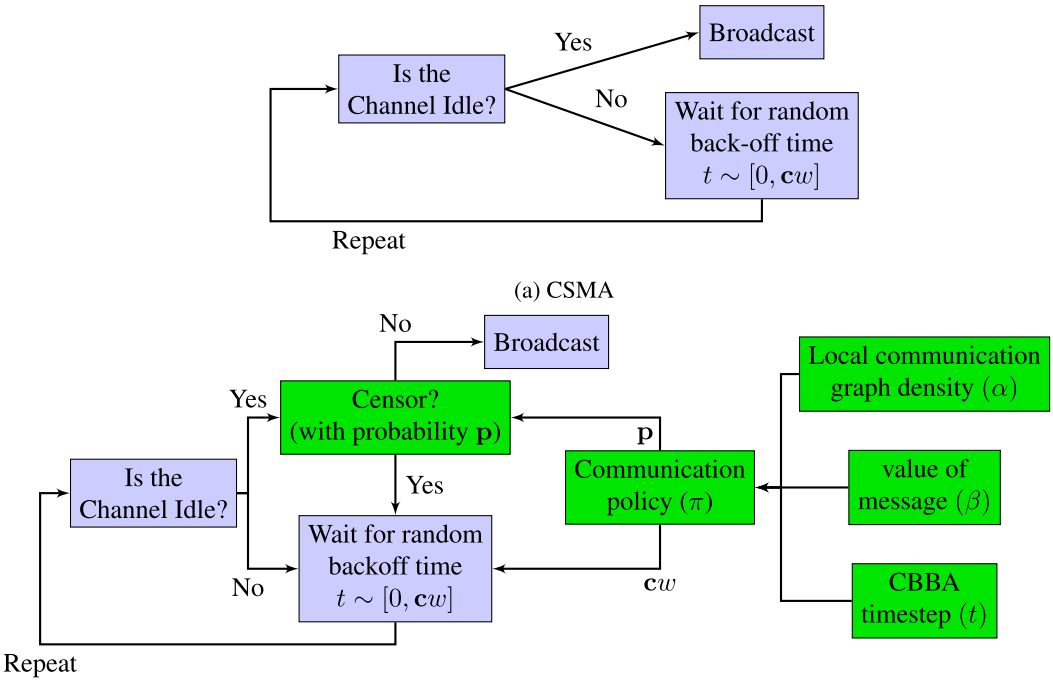


FIGURE 1. **(a) CSMA/CA scheme with fixed contention window size (cw). (b) shows the proposed version of CSMA that allows for censoring and priority scheduling based on a learned communication policy (attention to the green boxes that has been added to original CSMA (purple boxes) in design).**

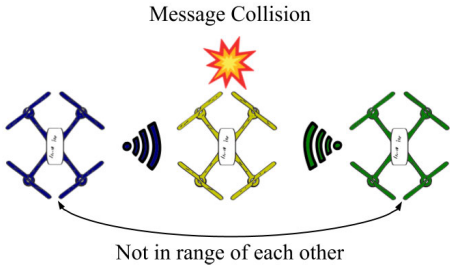


FIGURE 2. **Hidden node problem. Two agents (blue and green), not in sensing range of each other, broadcast message at the same time resulting in a "message collision". The information carried by these two messages cannot be recovered by yellow agent.**

consensus times for CBBA under TCP and UDP unicast mode are far longer compared to UDP broadcast.

Fig. 1a shows the working of CSMA/CA algorithm. Prior to transmission, nodes sense the medium for traffic. If the medium is found to be busy, the transmission is deferred for a random interval chosen from a fixed contention window size (generally chosen to be 16). This random interval reduces the likelihood of two or more nodes waiting to broadcast to start transmitting immediately upon termination of the current transmission, effectively reducing the incidence of collision. Running CSMA ensures that an agent remains silent when one of its neighbors is broadcasting. However, agents in the second neighborhood[1](hereinafter referred to as 2-hop neighbors) of the broadcasting agent cannot sense this traffic and can choose to broadcast at the same time, leading to message collision at a common neighbor. This is commonly referred to as the *hidden node problem* (depicted in Fig. 2 for only 3 agents). Hidden node problem results in packet loss; slowing down the conflict resolution process of a consensus algorithm such as CBBA. Although handshaking techniques can reduce the hidden node problem [53], they could introduce the latency in the ad-hoc network which is not desirable in time-critical task allocation problems such as search and rescue.

Bandwidth limitation is another constraint encountered when implementing a message-intensive algorithm like CBBA. Whenever an agent takes control of the medium for broadcasting, its neighbors have to wait. Under a random scheduling mechanism for broadcasting offered in CSMA, agents with no useful information can take control of the medium fairly often, blocking its neighbors with potentially useful information leading to a slower conflict resolution process.

[1]Second neighborhood of a node is the vertex-induced subgraph of all nodes that are at distance two (also known as 2-hop) from the current node.

IV. PROPOSED APPROACH

The main contribution of this work is to modify CSMA by adding a new communication policy to address the communication limitations, *i.e.* limited bandwidth and message collision. As explained later, we design a RL framework to learn the mentioned communication policy that maximizes the task allocation reward (see Fig. 1b, green blocks are added as new components to the original CSMA). This section discusses different components of the communication policy namely, censoring and scheduling along with the associated features.

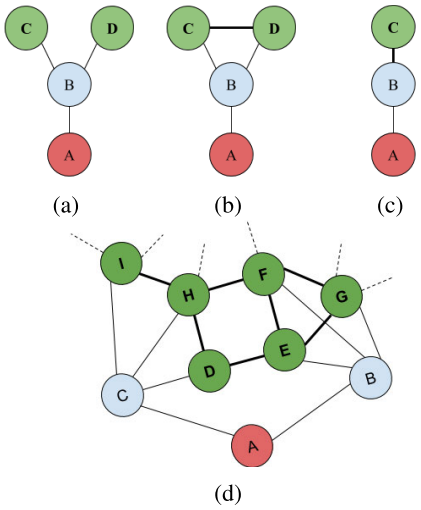


FIGURE 3. Different communication graph cases. Since agents C and D are connected in (b), only one of them can be scheduled to broadcast at the same time as agent A. This makes the network density experienced by agent A in (a) different from that in (b), even though agent A has 2 nodes in its 2-hop neighborhood for both. However, network density for A in (b) and (c) are similar. (d) shows agent A in a larger communication graph with its 2-hop neighbors shown in green. These green nodes along with the edges between them (shown in thick lines) constitute the second neighborhood of A. {G, D, I}, {G, H}, {E, H}, {E, I} and {F, D, I} are the five possible maximal independent sets of the second neighborhood of A.

A. CENSORING BASED ON LOCAL COMMUNICATION GRAPH DENSITY

The idea of censoring messages to address communication constraints have been explored in previous works. Ref. [38] uses censoring in distributed sensing problem. Inspired by this, we alter vanilla CSMA to allow for self-censoring of agents. Under this modified version of CSMA, agents choose to censor themselves with a probability **p** when they have an opportunity to broadcast. Whenever an agent censors itself, it effectively gives an opportunity for other agents in its 2-hop (or hidden nodes) to broadcast; reducing the likelihood of message collision. Since an agent cannot detect when its hidden node broadcasts, this censoring has to be random. We hypothesize that random self-censoring of agents with optimal censoring probability should improve the overall throughput of the network by reducing message collisions thereby improving the conflict resolution process of CBBA. Fig. 1b shows the modified CSMA protocol with censoring probability **p**.

Intuition suggests that optimal censoring probability must depend on the *crowdedness* (or density) of an agent's local communication graph. Furthermore, it is clear that an agent with an empty second neighborhood does not need to censor itself as there are no hidden nodes. However, an agent with ten 2-hop neighbors must censor itself with higher probability to give a fair chance at broadcasting for each of its ten potential hidden nodes. One way to represent this density is to simply count the 2-hop nodes, but there is a caveat to this approach. Not all 2-hop nodes can behave as hidden nodes at the same time.

Consider, agent A in the first three cases shown in Fig. 3. In Fig. 3a and Fig. 3b, there are two agents (C, D) in the 2-hop neighborhood. However, when running CSMA, only one agent (either C or D) can be scheduled to broadcast at the same time as agent A in case 3b. This is due to the fact that running CSMA ensures that neighbors are not scheduled at the same time. Therefore, the network density feature of agent A in Fig. 3b and Fig. 3c should be the same and different from that of agent A in Fig. 3a. Any network density feature must account for this redundancy in counting that occurs due to edges between 2-hop nodes.

To take into account this redundancy and inspired by [54], we consider maximal independent set of the second neighborhood of the agent, instead of its hidden nodes. In fact, each maximal independent set of an agent in the second hop neighborhood represents a distinct combination of hidden nodes. Thus, the expected size of an agent's maximal independent set would be a better measure of the local communication graph density. Assuming the graph is connected, maximal independent sets can be found by listing all the maximal cliques of its complementary graph, which is a well-studied problem in graph theory [55]. We use the Bron-Kerbosch (BK) algorithm [55], which is a recursive backtracking algorithm, to list all maximal cliques of the complementary second neighborhood. The worst-case running time for BK algorithm on a graph with n vertices is $\mathcal{O}(3^{n/3})$. Although this theoretical bound is non-polynomial, experience has shown that it is much faster in practice [56]. Furthermore, the probability of each of these maximal independent sets cannot be computed unless the entire graph is known locally, which would involve huge communication cost for larger graphs. Therefore, we use average cardinality of maximal independent sets as a measure of the local graph density referred as *BK feature*. Let the second neighborhood of agent i be a graph G_i with n maximal independent sets $\{m_1, \ldots, m_n\}$, then the *BK feature* is defined as:

$$\alpha^i = \frac{1}{n}\sum_{j=1}^{n} |m_j|. \tag{2}$$

Fig. 3d shows an example where {G, D, I}, {G, H}, {E, H}, {E, I} and {F, D, I} are the five possible maximal independent sets of second neighborhood of agent A. Each set represents a combination of agents that can be scheduled at the same time as agent A under CSMA. The BK feature of agent A is 2.4. Each message sent from agent j and received by agent i contains two sets of information: bidding information of agent j and j's neighbor list $\mathcal{N}_j$, the latter is essentially used to construct the second neighborhood graph locally at the start of CBBA as shown in lines 5 - 7 of Algorithm 1. In Section V-C1, we show that the BK feature can be used to efficiently censor agents when compared to a simple 2-hop count.

B. SCHEDULING BASED ON VALUE OF MESSAGE

In decentralized algorithms such as CBBA, it is likely that only a few agents will have valuable information at any given time. Vanilla CSMA algorithm treats all the agents

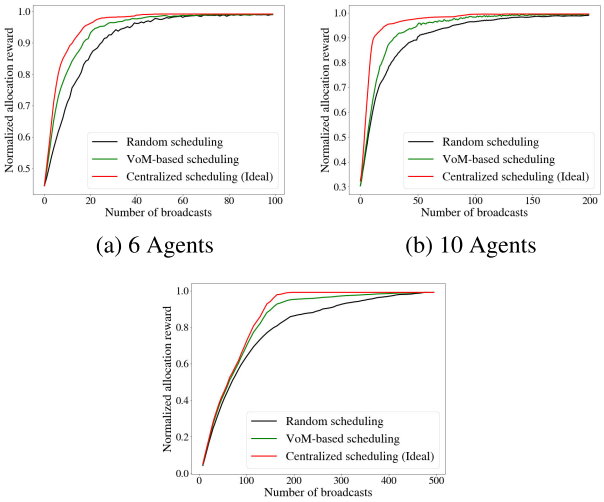
with the same priority limiting bandwidth for other agents with potentially important messages to broadcast. Priority based scheduling in CSMA has been explored in previous work [57], [58] by setting a lower contention window size for agents with higher priority. In our work, we prioritize agents based on the value of their message towards the team's final goal of conflict resolution. We now focus on defining a value of message (VoM) metric for a message in CBBA. A key challenge here is to keep this metric local, that is no additional communication must happen for computing this metric, as it might defeat the original purpose of limiting communication. Let m_t^i denote the message sent by agent i at time t. In ideal scenario where there are no message collisions, this message is received by each of its neighbors and is used in the conflict resolution process based on the rules described in conflict resolution phase of CBBA. The intuition here is that if message m_t^i from agent i does not alter the winning bids list, $\boldsymbol{y}$, of any of its neighbors, it is uninformative and its value must be 0. If we denote, the new winning bid list of a neighboring agent j after resolving conflicts using message m_t^i as $\boldsymbol{y}_j^{m_t^i}$ and the neighborhood of agent i as $\mathcal{N}_i$, then the value of message m_t^i could be written as,

$$\beta_t^i = |\mathbf{N}_i|^{-1} \sum_{j \in \mathbf{N}_i} \left\| \text{sign}\{\mathbf{y}_j^{m_t^i} - \mathbf{y}_j\} \right\|_1. \tag{3}$$

However, agent i does not have access to the current winning bid list of any of its neighbors, and any means of obtaining this would incur more communication cost. Therefore, we use the most recent message received from agent j as surrogate to $\boldsymbol{y}_j$. We denote this recent message as $\bar{\mathbf{y}}_j$. By making this temporal approximation, we are able to estimate value of a message locally without additional communication. With this change, (3) can be approximated as,

$$\beta_t^i \approx |\mathbf{N}_i|^{-1} \sum_{j \in \mathbf{N}_i} \left\| \text{sign}\{\bar{\mathbf{y}}_j^{m_t^i} - \bar{\mathbf{y}}_j\} \right\|_1. \tag{4}$$

To illustrate the effectiveness of our VoM feature in capturing the true value of a message, we investigate a simple experiment. We consider agents in an ideal communication environment without any message collisions and implement three different scheduling algorithms - random scheduling, scheduling based on VoM defined in (3) and centralized scheduling based on (4). Under centralized scheduling, agents have access to the state vectors of their neighbors and hence can make an exact prediction of the value of a message. Although, centralized scheduling is unrealistic, it is the theoretical upper bound on how fast the conflict resolution process could be. Fig. 4 shows that VoM-based scheduling, with no additional communication, outperforms random scheduling and is comparable to centralized scheduling. This feature is computed at every timestep of CBBA, as shown in line 11 of Alg. 1 and Section V-C2 shows how this metric is used by agents to increase their priority (in a decentralized manner) when running CSMA, resulting in faster conflict resolution process.



(a) 6 Agents (b) 10 Agents

(c) 100 Agents

FIGURE 4. VoM-based scheduling. Under centralized scheduling (ideal unrealistic case), agents have access to state vectors of their neighbors. VoM-based scheduling (green), without additional communication, can schedule as effectively as centralized scheduling (red) and outperform random scheduling (black) found in vanilla CSMA.

Algorithm 1 Learning to Censor and Schedule

```
Initialize actor (φ), critic (θ) parameters, replay buffer D
for N̂ episodes do
    Initialize task allocation problem
    for each agent i do
        Broadcast neighbor list by 3-hop communication
        Reconstruct second neighborhood
        Calculate local network density α^i using Eq. 2
    end for
    for t time steps do
        Run Task selection phase of CBBA [1]
        Calculate value of message β_t^i using Eq. 4
        Set CSMA parameters as a_t^i = π(o_t^i) + N(0, σ^2)
        Broadcast messages over the network
        Run conflict resolution phase of CBBA [1]
        Observe reward r_i^t and next observation o_{t+1}^i
        Store (o_t, a_t, r_t, o_{t+1}) in replay buffer D
    end for
    for each gradient step do
        Sample minibatch (o^j, a^j, r^j, o'^j) from buffer D
        Update φ and θ based on MADDPG [59]
    end for
end for
```

C. LEARNING TO CENSOR AND SCHEDULE

We are interested in finding an optimal, decentralized cooperative policy that uses the above features to speed up conflict resolution process. This decision making problem can be formalized as a *Multiagent Markov Decision Process* (MMDP) [60] which is a tuple,

$$\mathbf{M} = \left\langle \mathbf{S}, \mathcal{I}, \{\mathcal{U}_i\}_{i \in \mathcal{I}}, \pi, \mathcal{R}, \mathcal{T} \right\rangle$$

where $\mathbf{S}, \mathcal{I}$ are the set of states and agents. In decentralized setting, each agent i receives an observation o_t^i that is correlated with the current state of the team s_t. Agent i performs an action $a_t^i \in \mathcal{U}_i$, sampled according to the policy $\pi^i(o_t^i; \phi^i)$

parameterized by ϕ^i. The joint action $\boldsymbol{a}_t = \{a_t^1, \ldots, a_t^n\}$ transitions the current state s_t to next state s_{t+1} according to transition function $\mathcal{T}(s_{t+1}|s_t, \boldsymbol{a}_t)$ and each agent i makes a new observation o_{t+1}^i correlated with the new state s_{t+1}. The team receives a joint reward $r_t = \mathcal{R}(s_t, \boldsymbol{a}_t)$ defined in Eq.1. The goal is to find policy $\pi^i, \forall i \in \mathcal{I}$ such that the expected joint reward is maximized.

In environments with complex dynamics $(\mathcal{T}, \mathcal{R})$ like ours, model-free reinforcement learning frameworks which treat the environment as a black box are often used. Actor-critic algorithms, a class of model-free RL algorithms approximate expected reward by learning an action-value critic, $Q_i(o_t^i, a_t^i; \theta^i)$. In our work, we use MADDPG [59], a popular multiagent actor-critic algorithm to learn the optimal policy. MADDPG uses a centralized critic to address non-stationarity issues inherent to multiagent RL settings. The actor or the learnt policy uses only local observations; allowing for decentralized evaluation. Additionally, we allow parameter sharing for both actor and critic functions across the team due to homogeneous nature of agents in our problem. This simplification speeds up the training process and allows policy to generalize across different team sizes.

Fig. 5 illustrates the training process. In our case, the observation consists of three features, namely local communication graph density α_t^i, value of message β_t^i. and the normalized CBBA timestep $\bar{t}$. The action space consists of censoring probability p_i^t and contention window size cw_i^t:

$$o_i^t = [\alpha_t^i, \beta_t^i, \bar{t}], \quad a_i^t \in [0, 1]^2,$$
$$p_i^t = a_i^t[0], \quad cw_i^t = \left\lceil 16 a_i^t[1] \right\rceil,$$

where $\lceil . \rceil$ denotes the ceiling function and $a[i]$ denotes the ith dimension of a. Since the communication graph for a given episode is fixed, α_t^i is constant for all t within an episode. The change in CBBA reward from the last time step is used as the common reward (r_i^t) for learning. Gaussian noise $(\sigma = 0.1)$ is added to the action during learning to enable exploration of the policy space as shown in line 12 of Algorithm 1. The algorithm for learning optimal cooperative communication policy is summarized in Algorithm 1.

V. RESULTS

In this section, we evaluate the performance of the proposed algorithm (CA-CBBA) against different baselines. Application of the algorithm to two different scenarios is presented. Additionally, we perform ablation study to understand the contribution of each of the two components of the algorithm - censoring and scheduling. We also examine the generalizability of the learned communication policy to teams of different sizes allocating different task numbers.

A. IMPLEMENTATION DETAILS

The simulation environment consists of multiple agents performing decentralized task allocation in an in-house network simulator running UDP with IEEE 802.11b broadcast in ad-hoc settings. Agents use a modified version of CSMA/CA scheme shown in Fig. 1b with parameters defined by the learned communication policy.

1) BASELINE

In addition to CA-CBBA, we compare our algorithm to four other baselines defined below:

1) **CBBA-Ideal** - This consists of agents running CBBA without any communication constraints. All the agents in the team broadcast their messages to their neighbors and message collisions are ignored. This is an ideal scenario and represents the upper bound on the performance of any communication-aware task allocation algorithm.
2) **CBBA-Baseline** - This consists of agents running CBBA under communication constraints. Agents use CSMA to access the shared medium and broadcast their messages to neighbors. The convergence rate of CBBA-Baseline is limited by communication constraints.
3) **ACBBA** - Asynchronous version of CBBA presented in [3] which allows the agents independently rebroadcasting or refuse to broadcasting their message given the received messages from other agents. ACBBA shows more robust performance compared to CBBA in environments with imperfect communications.
4) **MCDGA** - In this approach agents either unicast (to a specific agent) or broadcast (to neighbors) their messages based on the local communication rules defined by the algorithm. We use a handshaking mechanism to unicast messages and hence this algorithm doesn't suffer from message interference like the other baselines and CA-CBBA. The downside of such a setup is that it takes longer to disseminate information.

2) TRAINING SETUP AND EVALUATION METRICS

We trained our algorithm on 6, 10 and 20 agent teams. In each training episode, agents are initialized in an arbitrary connected communication graph and perform specified rounds of CBBA. The experience tuple which consists of observation, action and reward is stored in a memory buffer. Batches of size 128 sampled from this replay buffer $\mathcal{D}$ are used to perform gradient updates on the communication policy represented using a feed forward neural network of depth 2 and width 64. In total, the actor and critic converge after 15, 000 and 20, 000 episodes for 6 and 10 agents case respectively. The learned policies were evaluated for 200 episodes for each of the following experiments. The average and variance of the following evaluation metrics are reported.

1) **Normalized CBBA reward** - Ratio of the task allocation reward of the entire team obtained at the end of running CBBA (or MCDGA) to the reward obtained from Centralized Sequential Greedy Algorithm (SGA).
2) **Fraction of conflicts** - Ratio of number of tasks that are allocated to multiple agents (conflicts) to the total number of tasks.

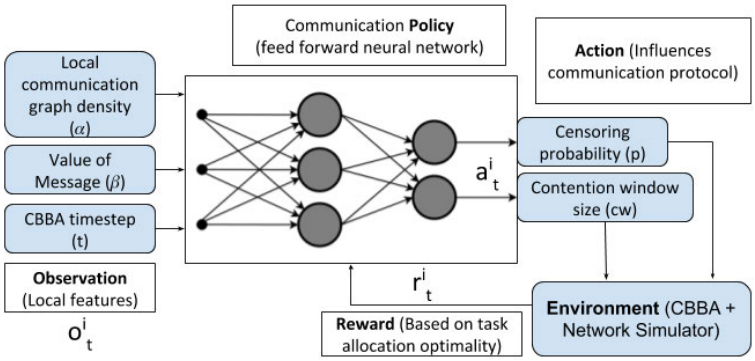


FIGURE 5. RL training process. Shown here is policy of a single agent parameterized by a neural network. The policy learns to control CSMA parameters - censoring probability and contention window size based on the input features described in Sections IV-A and IV-B.

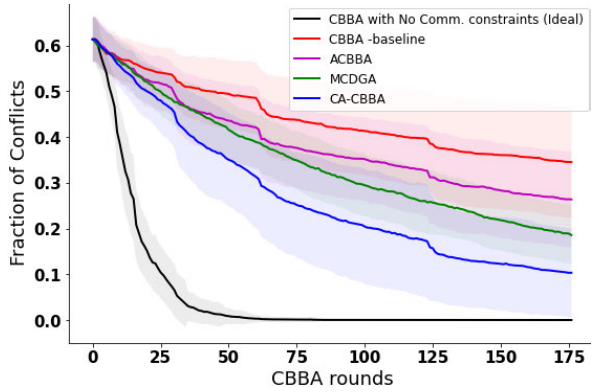


FIGURE 6. Fraction of conflicts vs rounds of communication in task allocation, for a team of 10 agents. The shaded region represents the standard deviation for each algorithm obtained from 100 different runs. CBBA-Ideal has the fastest conflict resolution but it represents an unrealistic scenario without communication constraints. Among the realistic algorithms, CA-CBBA and CBBA-Baseline and ACBBA use broadcasting setup while MCDGA uses a secure unicasting mode for communication. CA-CBBA performs better than the rest as it is able to efficiently censor and schedule agents to reduce message interference and use its bandwidth efficiently. Although, MCDGA does not suffer from message interference, it has a slower conflict resolution rate compared to CA-CBBA as unicasting is inefficient in disseminating information in a consensus setup.

3) **Convergence time** - Time taken for CBBA to converge to a conflict free allocation. This metric is defined in terms of rounds of CBBA or rounds of communication. Note: Every round of CBBA includes bundle building phase and consensus phases in which agents go through one round of communication.
4) **Fraction of Unassigned Tasks** - The fraction of unassigned tasks, *i.e.* the ratio of unassigned tasks to the total number of tasks, is an evaluation metric used in time-sensitive applications.

B. CONVERGENCE ANALYSIS

In this subsection, we compare the performance of CA-CBBA against different baselines described above.

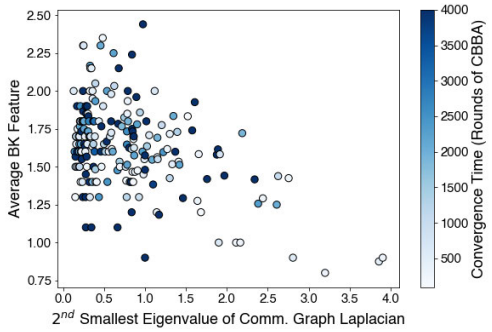
1) CONFLICT RESOLUTION

Conflict resolution is an important aspect of decentralized task allocation. Communication constraints hinder the message passing and result in slower conflict resolution. Fig. 6 compares conflict resolution ability of CA-CBBA and other baselines by plotting the fraction of conflicts across the team against rounds of communication. Since, CBBA-Ideal ignores communication constraints it is unrealistic. However, its conflict resolution curve acts as the upper bound on performance for other algorithms. CBBA-Baseline is the vanilla application of CBBA with realistic communication constraints and hence has the slowest conflict resolution rate.

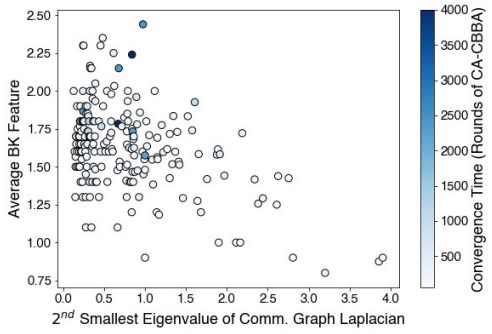
MCDGA and CA-CBBA (our algorithm) take two different approaches to resolve communication constraints. MCDGA prunes unwanted broadcasting of messages in original CBBA by modifying the message. This modification forces MCDGA to unicast messages in certain cases. The advantage of this approach is that messages can be transmitted reliably. However, unicasting is an inefficient form of communication for consensus algorithms as information is disseminated slower. ACBBA considers rebroadcasting some messages or censoring to broadcast, therefore it requires smaller number of messages compared to CBBA, but it suffers from hidden node problems. On the other hand, CA-CBBA uses broadcasting but deals with the communication constraints by censoring and scheduling agents based on the Value of their Message. This allows CA-CBBA to resolve conflicts faster than other baselines that operate under communication constraints.

2) THE EFFECT OF COMMUNICATION GRAPH ON CONVERGENCE TIME

Communication constraints do not impact all the CBBA runs equally. CBBA runs with very less connectivity (or larger number of hidden nodes) suffer from message interference. Fig. 7a shows the convergence time of CBBA-Baseline for different runs where each run is characterized by the average BK feature of the nodes, as an indicative of the network hidden nodes density, and second smallest eigenvalue of Graph Laplacian of the corresponding communication graph, as the indicative of graph connectivity [61]. It can be seen that CBBA-Baseline runs that are close to the top left portion of the graph are timed out (*i.e.* it takes longer



(a) Convergence time for CBBA-Baseline



(b) Convergence time for CA-CBBA

FIGURE 7. Convergence time of CBBA and CA-CBBA for 10 agent teams. Each point represent one of the 200 different runs made on a random communication graph. Two features for each communication graph - the average BK feature value of the nodes and the second smallest Eigenvalue of graph Laplacian are shown. It can be seen that most runs in the upper-left part of Fig. 7a gets timed out as messages in these runs are not broadcasted reliably due to communication constraints. CA-CBBA alleviates this issue.

than 4000 rounds to converge) compared to the ones on the bottom right. CA-CBBA is able to alleviate communication issues in such CBBA runs by efficiently censoring as evident in Fig. 7b, resulting in converging faster compared to CBBA-baseline. Note that the darker circles are associated with slower convergence.

3) THE EFFECT OF TASK NUMBER ON CONVERGENCE TIME

Fig. 8 shows the box plot of the convergence time of CA-CBBA and different baselines for different task numbers. It can be seen that convergence time for all runs of CA-CBBA is better than 75 percentile runs of CBBA-Baseline. Convergence times for a very few runs of CBBA-Baseline is less than CA-CBBA. These correspond to task allocation runs with easier (less prone to message interference) communication graphs such as the ones found in bottom right of Fig. 7a. Furthermore, CA-CBBA is able to consistently outperform CBBA-Baseline for a wide range of task numbers even though the communication policy was trained on a specific task number of 100.

C. ABLATION STUDY

An ablation study is performed to show the contribution of censoring and scheduling to the performance of CA-CBBA.

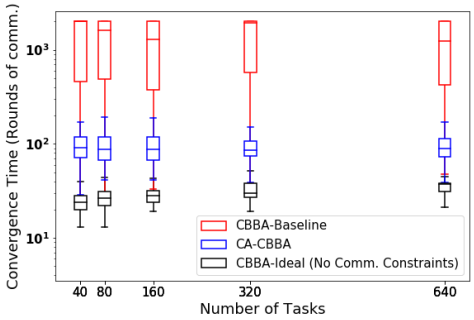


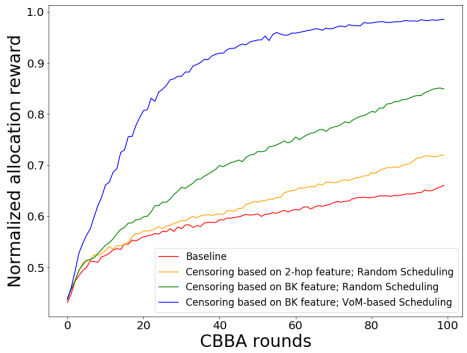
FIGURE 8. Convergence time (in rounds of communication) vs Number of tasks for a 6 agent team. CA-CBBA consistently outperforms CBBA-Baseline on experiments with wide range of task number even though the policy was trained on a specific task number of 100. It should be noted that CA-CBBA and CBBA-Baseline box plots overlap due to the presence of graphs such as fully-connected and long-chain that are inherently easier and harder to run task allocation on and must not be interpreted to mean that the results are not statistically significant. When controlled for such variations in communication graph, CA-CBBA outperforms other baselines as shown in Fig. 7.

1) REDUCED MESSAGE COLLISIONS DUE TO EFFICIENT CENSORING

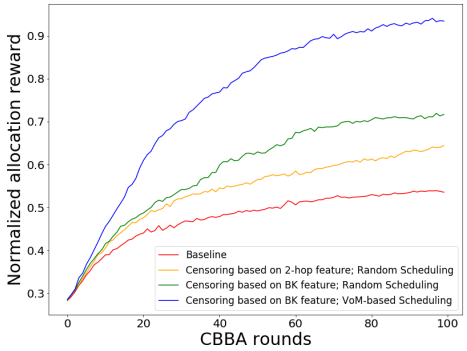
We train two altered versions of the algorithm. The first one uses 2-hop count for local communication graph density while the second one uses Bron-Kerbosch feature. Both these versions do not use Value of Message and instead, use the default random scheduling found in Vanilla CSMA. This is done to study only the effect of censoring in isolation. As discussed in Section IV-A, 2-hop count cannot capture the local network density across multiple communication graphs as effectively as Bron-Kerbosch feature. Hence, the policy learned using 2-hop count cannot censor agents efficiently to reduce message collisions. This is evident from the slower conflict resolution for 2-hop count based censoring (orange) compared to BK feature based censoring (green) in Fig. 9a and Fig. 9b. Furthermore, the claim is supported by calculating average throughput shown in Table 1. The average throughput (successful message transmissions) is higher for BK feature case compared to 2-hop count as a result of efficient censoring. Both these censoring methods outperform throughput in baseline case.

2) FASTER CONFLICT RESOLUTION DUE TO VoM-BASED SCHEDULING

We have shown censoring of messages increases throughput by addressing the hidden node problem. However, bandwidth limitation in constrained communication, cannot be addressed by censoring. This section introduces a new component for efficient scheduling of communication to address the bandwidth limit. This is motivated by this fact that the random scheduling causes agents with important messages to wait until other agents finish their transmission, slowing down the conflict resolution process. This is because random scheduling is inherently blind to the value of an agent's message towards the final goal of the team. We address this issue in our algorithm by prioritized scheduling based on the value of message. Fig. 9a and Fig. 9b show that the



(a) Six agents



(b) Ten agents

FIGURE 9. Variations of CA-CBBA that only censor based on 2-hop count (orange) and BK feature (green) perform better than baseline (red). BK feature censors agents efficiently compared to 2-hop count as it can express local graph density better. CA-CBBA (blue) performs the best as a result of efficient censoring and prioritized scheduling that allows informative agents to use the bandwidth effectively.

conflict resolution process for our proposed algorithm (BK feature-based censoring and VoM-based scheduling) (blue) is faster compared to the two learning versions that use random scheduling and the baseline (red). Furthermore, Table 1 shows that throughput for BK feature based censoring with 2-hop count based scheduling (green) and BK feature based censoring with VoM based scheduling (blue) are the same even though they show much different conflict resolution curves in Fig. 9a and Fig. 9b. This shows that both algorithms censor agents equally efficiently to allow higher throughput, but the information content of the sent messages are higher on average in our VoM-based approach, which leads to a faster conflict resolution process.

D. COMMUNICATION POLICY GENERALIZES TO DIFFERENT TEAM SIZES

Since the communication effects of message collision and bandwidth limitations are local, it should be possible to learn a policy that can generalize to any team size. By constructing features that are local (and depend only on neighboring agents) and by sharing parameters of the policy, we are, in fact, able to learn optimal policy that generalizes well to larger team sizes. Fig. 10 shows the policy learned using 6 agents (solid blue) and a policy learned with 10 agents (dashed blue) along with the case where there is no communication constraints during CBBA execution and we call it ideal case (black). Note that in this graph the *y-axis* is the task allocation reward normalized to ideal case (the maximum possible reward), which means the maximum value of *y-axis*is one. We expect the task allocation reward for all algorithms increases as CBBA is executed (*x-axis* shows the round of communications in CBBA). However, the convergence rate the reward could be different for different baselines. The graph shows learned communication policy improves the convergence rate of task allocation reward compared to the no-learned policy. The ideal curve cannot be attained under any real communication constraints, so it provides a (possibly very optimistic) upper bound on performance (with the no learning approach providing a lower bound). The key point here is that, when evaluated on a 10-agent team, the 6-agent policy (solid blue) matches quite closely to the performance of the 10 agent policy, showing that the learned result generalizes well to a larger team. Fig. 10b shows that CA-CBBA is able to generalize to 20 agents, when the policy is trained for 10 agents. For larger networks (here 20 agents) non-stationarity dominates and the MADDPG algorithm fails to learn. However, we can evaluate the performance of the 6-agent and 10-agent policies, which again show similar conflict resolution curves in those settings and out-perform the no learning baseline. These results suggest that the policy learned from smaller teams can be used on larger teams for which it is hard/impossible to learn the optimal policy.

E. VALIDATION IN A HIGH-FIDELITY NETWORK SIMULATOR

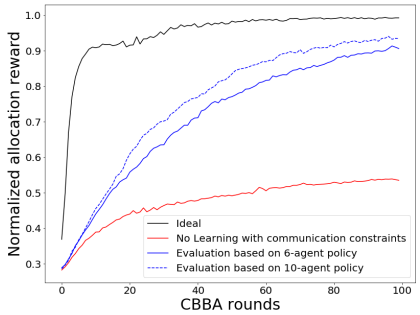
We used a simple in-house simulator written in Python for training the policy because it is easier to interface with the DRL frameworks used for MARL. This simulator models key aspects of realistic communication networks, such as message collisions and bandwidth interference. However, it is also conservative in its treatment of communication constraints. For example, message delivery in realistic networks is not a deterministic process with a cut-off distance (as modelled by the disc model used in the *Simple* simulator). It is a stochastic process with the probability of delivery proportional to Bit Error Rate (BER) which in turn depends on the Signal to Interference Noise Ratio (SINR) at the receiver. This would result in nodes outside the disc to receive messages occasionally.

This subsection validates our algorithm on a realistic network simulator based on NetSim [62] that considers a packet erasure channel and models the probability of delivery of a packet using Bit Error Rate (BER). We call this the *BER* simulator, and the key equations are as follows. Let the probability of delivery of a message broadcast by agent i at agent j be denoted as β_{ij} and let b_{ij} be the BER of the received message of size N

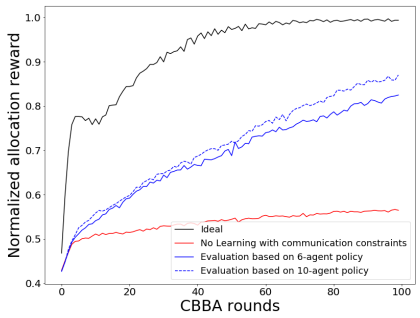
$$\beta_{ij} = 1 - (1 - b_{ij})^N. \tag{5}$$

TABLE 1. Increased throughput due to efficient censoring. Bron-Kerbosch (BK) feature captures the local network density better than a simple 2-hop count. Therefore, the policy learned using Bron-Kerbosch feature results in reduced message collisions and higher throughput.

Throughput	CBBA-Baseline	2-hop count Censoring; Random Scheduling	BK feature Censoring; Random Scheduling	CA-CBBA
6 agents	185.61 ± 8.22	196.32 ± 7.74	212.55 ± 7.65	212.45 ± 7.58
10 agents	248.45 ± 12.25	265.36 ± 12.20	289.30 ± 12.05	289.90 ± 12.10



(a) 10 Agents



(b) 20 Agents

FIGURE 10. Generalization of learned policies to different team sizes. Fig. 10a shows that the policy learned using 6-agent team (solid blue) performs as good as 10-agent policy (dashed blue). Fig. 10b policy learned with smaller teams perform reasonably, outperforming the baseline (red) for larger team sizes (20 agents here) where it can be hard for MADDPG to learn.

Under QPSK modulation, b_{ij} can be related to the power of the message received P_{ij} and the power of the interference noise P_n as,

$$b_{ij} = \text{erfc}\left(\sqrt{\frac{P_{ij}}{P_n}}\right), \tag{6}$$

where erfc$(\cdot)$ is the complementary error function and P_n includes thermal noise and interference from hidden nodes in the case of UDP communication. Furthermore, if the transmission power of agent i is P_i,

$$P_{ij} = P_i - P_l, \tag{7}$$

where P_l is the loss in power due to different physical phenomenon such as path loss and fading loss which is a function of the distance between transmitter and receiver among other parameters. In the *BER* simulator, we use Hata-Okumura model [63], a standard loss model for broadcast in dense environments. In *BER* simulator, the probability of two messages colliding (and getting rejected) depends on the interval of overlap between the messages. Under the simple simulator, a harsher condition that ignored both the messages completely even if this interval is very small was imposed. The evaluation results of our algorithm for 6 and 10 agent teams on both the *Simple* and *BER* simulator is shown in Table 2. Results demonstrate our learning framework is able to generalize to the complex and realistic network simulator further supporting the case for the features crafted in Section IV. Furthermore, it also shows that the modified simulator is less conservative in its predictions compared to the simple simulator.

F. APPLICATIONS

We have evaluated our algorithm and other baselines in two different applications of task allocations: search and rescue and task scheduling for time-limited tasks.

1) TIME-SENSITIVE TASK ALLOCATION: SEARCH AND RESCUE

In many applications, tasks are time-sensitive, i.e, they have a higher reward if done earlier. For instance, in a search and rescue scenario, a group of agents (vehicles) should move to the assigned targets and provide medical supplies, food, or provide transportation as soon as possible. In these cases, the tasks that are time-critical must be serviced earlier. This can be accounted for by using a time-discounted reward [1], where the total reward of servicing the tasks assigned to agent i is

$$S_i^{\psi_i} = \sum_{j=1}^{N_t} \lambda^{\tau_i^j(\psi_i)} \bar{c}_j, \tag{8}$$

$\lambda \in (0, 1]$ is a discount factor and $\tau_j^i(\psi_i)$ is the time when agent i arrives at task j along the allocated path ψ_i. Also $\bar{c}_j$ is the static reward considered for task j and typically implies the task deadline d_j. We use the exponential function to compute the task reward

$$\bar{c}_j = e^{-d_j}. \tag{9}$$

Assume that d_j is the distance traveled by agent i to arrive at target j, each vehicle has speed v_i, and target j is serviced in the order of $\zeta_j \in \{1, 2, \ldots, L_t\}$. We further assume that the time required for servicing each target is δ, then $\tau_j^i(\psi_i)$ is

$$\tau_j^i(\psi_i) = d_j {v_i}^{-1} + (\zeta_j - 1)\delta \tag{10}$$

We consider a search and rescue experiment where the vehicles start from a fixed location and the tasks are randomly located in the environment (see Fig. 12). For this

TABLE 2. Normalized task allocation reward after 50 rounds of CBBA in *Simple* and *BER* simulator. The policy is able to improve performance in both simulators, showing that features defined in section IV are general enough to capture the fundamental constraints posed by real communication networks.

	Baseline (*Simple* Simulator)	Baseline (*BER* Simulator)	Learned policy (*Simple* Simulator)	Learned policy (*BER* Simulator)
6 agents	0.59	0.67	0.95	0.97
10 agents	0.47	0.58	0.83	0.89

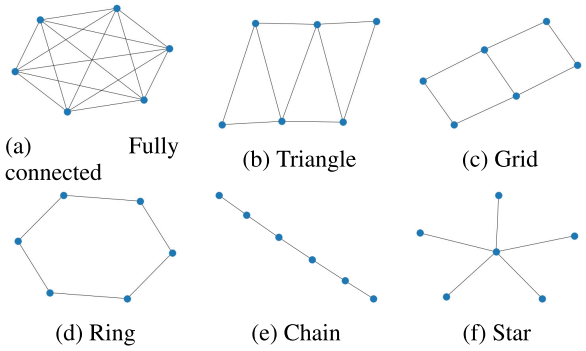


FIGURE 11. Different graph typologies used for search and rescue experiment.

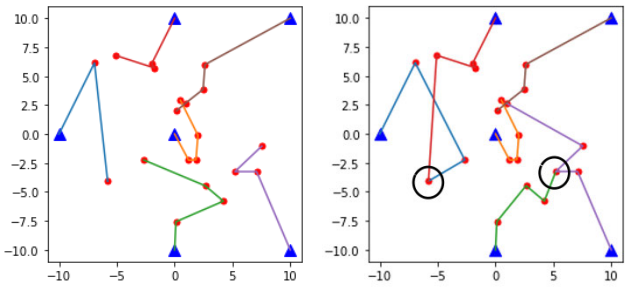
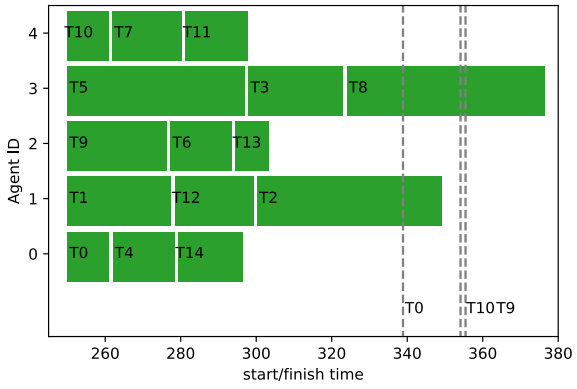
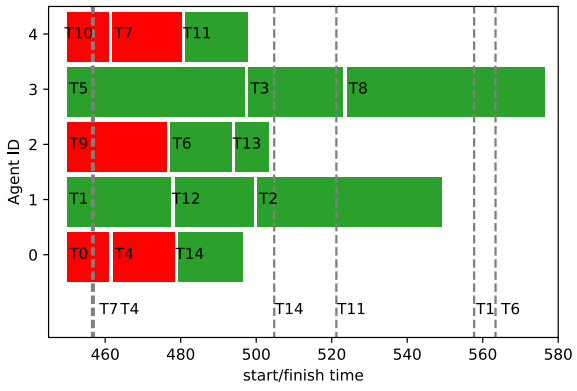


FIGURE 12. Search and rescue experiments with 6 vehicles (blue triangles), twenty survivors are located randomly (red circles), and are served in the order of their location in the assigned path. The task assignment is stopped after 175 rounds of communications and then the agents are served the tasks according to the order in the assigned path. While CA-CBBA and ideal case achieves a conflict-free task assignment in (a), original CBBA that does not consider the communication constraints does not converge and two of the tasks are conflicted between two agents (shown in black circles).



(a) CA-CBBA task-time schedule



(b) MCDGA task-time schedule

FIGURE 13. Time scheduling for the task *T0* to *T14* is shown when CA-CBBA (left) or MCDGA (right) is applied for task scheduling, for five agents in a triangle communication topology. Only tasks deadlines lying in the current time window is shown by dashed lines, the tasks which are finished after the deadline are shown in red and tasks that meet the deadline is marked by green.

analysis we just consider the initial task allocation problem using an imposed network of the types shown in Fig. 11) (plus a random one). The network is fixed during this initial task allocation. The results in Table 3 show that MCDGA performs the best for a star-shaped topology (requires fewer messages in Unicast communication), but is the worst for a chain graph (presents networks with longest diameter that inherently requires more communication messages in an unicast setting). On the other hand, CA-CBBA outperforms the other algorithms for most topologies (fully connected, triangle, chain, and random cases).

2) TIME SCHEDULING OF MULTIPLE TASKS

In this experiment agents should service multiple tasks, but the tasks have to be executed in the order of their deadlines to minimize the number of missing tasks. To compute the reward function for this task-assignment problem, we use Eq. 8 with two modifications:

1) A path indicates the order of accomplishing the tasks. When a task is accomplished, the agent starts next task in the path immediately. This means there is no physical distance between the tasks.
2) Each task has a specific level of difficulty, which implies the time required for accomplishing task would be different across different tasks. The time required to accomplish the task j by agent i is computed as:

$$\delta_j^i = l_j \gamma^i \tag{11}$$

where l_j refers to the level of difficulty of task j, (*i.e.*, more difficult task has higher values of l_j). Also γ^i

TABLE 3. Search and rescue experiments with N_u vehicles with different velocities in the range of $[0.5, 5] m/s$, the initial location of the vehicles are fixed in $[-10, 10] \times [-10, 10]$ (see Fig. 12 for 6 agents configurations), but the N_t tasks are located randomly in $[-10, 10] \times [-10, 10]$. Each agent can serve up to L_t tasks. The performance is reported in terms of the fraction of conflicted task and fraction of number of unassigned tasks (lower is better) for the CBBA [1], ACBBA [3], MCDGA [2], and CA-CBBA (our approach). As the number of agents increases, we expect longer time for the convergences due to communication constraints, but T_a is defined as the time the task assignment is stopped.

Topology	Algorithm	(N_u, N_t, L_t)-T_a					
		(4,7,2)-100		(6,16,3)-200		(9,42,5)-510	
		conflict	unassigned	conflict	unassigned	conflict	unassigned
fully-connected	CBBA	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
	ACBBA	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
	MCDGA	**0.0**	**0.0**	0.03	**0.0**	0.06	0.01
	CA-CBBA	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	0.01
triangle	CBBA	0.14	0.04	0.08	0.03	0.02	**0.0**
	ACBBA	0.07	0.02	0.01	**0.0**	0.01	**0.0**
	MCDGA	0.01	**0.0**	0.15	0.04	0.27	0.21
	CA-CBBA	**0.0**	**0.0**	**0.0**	**0.0**	**0.001**	**0.0**
grid	CBBA	0.25	0.10	0.24	0.17	0.14	0.07
	ACBBA	0.23	0.13	0.25	0.16	0.14	0.08
	MCDGA	**0.0**	**0.0**	0.06	**0.0**	0.15	0.08
	CA-CBBA	**0.0**	**0.0**	**0.005**	**0.0**	**0.0**	**0.0**
ring	CBBA	0.22	0.14	0.2	0.1	0.02	0.001
	ACBBA	0.23	0.10	0.16	0.07	0.005	**0.0**
	MCDGA	**0.0**	**0.0**	0.07	0.005	0.29	0.22
	CA-CBBA	**0.0**	0.002	**0.0**	**0.0**	**0.0**	**0.0**
chain	CBBA	0.14	0.04	0.14	0.03	0.13	0.08
	ACBBA	0.07	0.02	0.18	0.06	0.07	0.02
	MCDGA	0.01	**0.0**	0.27	0.15	0.36	0.30
	CA-CBBA	**0.0**	**0.0**	**0.005**	**0.0**	**0.003**	**0.0**
star	CBBA	0.3	0.27	0.30	0.48	0.24	0.52
	ACBBA	0.25	0.15	0.28	**0.31**	0.26	0.40
	MCDGA	**0.10**	0.06	**0.21**	0.34	**0.19**	**0.35**
	CA-CBBA	**0.07**	**0.08**	0.28	0.43	0.24	0.52
random	CBBA	0.03	0.03	0.11	0.06	0.16	0.11
	ACBBA	0.02	0.001	0.06	0.02	0.08	0.04
	MCDGA	0.04	0.008	0.11	0.05	0.25	0.19
	CA-CBBA	**0.01**	**0.003**	**0.002**	**0.0**	**0.02**	**0.007**

indicates the time the agent i requires to finish the task with level of difficulty equal to 1. Using this definition, Eq.10 is rewritten as

$$\tau_j^i(\psi_i) = \sum_{k \in b_i, o_k < o_j} \delta_k^i \tag{12}$$

where b_i is the set of task indices in agent i's bundle which are serviced before task j.

As before, we analyze the task allocation and execution process separately. First tasks are assigned between agents in a designated time (task assignment step). Then, each agent accomplishes the tasks in the assigned order (task accomplishment step). Allocating an appropriate time for task assignment is challenging as the convergence time for task assignment would be different for different task allocation algorithms. Furthermore, the task assignment is decentralized and the exact convergence time could not be retrieved without global knowledge. To overcome these issues, we use Monte Carlo sampling to estimate the required time for converging. We have run baselines for 100 trials and measure the average convergence time for each algorithm, *i.e.* when the task assignment is conflict-free and all the tasks are allocated. In this example we assume agents communicate via a triangle-shaped graph (see Fig. 11). The result from this Monte-Carlo sampling suggests the required 250, 450, and 1000 rounds of communication (roc) for the convergence of CA-CBBA, MCDGA, and CBBA-baseline respectively. We consider this required time for task assignment step. So, the accomplish task step starts in different times for each algorithm. We unify the time scales in two steps, such that task deadlines are randomly selected between 300 to 900 (roc). The agents speeds are randomly chosen from $\gamma \in [1, 10]$ and the level of difficulty for the tasks is chosen from $l \in [1, 4]$.

Fig. 13 shows the results of a task assignment of 15 tasks between 5 agents (each has the capacity of accomplishing three tasks). As the results show, CA-CBBA was able to finish all the tasks before their deadlines (**T0** has the earliest deadline at 339 rounds of communication and **T3** has the latest deadline of 876 rounds of communication). This result also confirms the final task assignment bundles for both MCDGA and CA-CBBA are the same, which is as expected because they are basically trying to optimize the same reward function, which prioritizes the tasks based on their deadlines.[2] This example shows MCDGA requires longer time to converge, which leads to miss five tasks. Note the task **T5** started before its deadline but it is not done by its deadline and considered as a missed task. In this example, CBBA-baseline requires 900 rounds of the communication to converge in average, which means it misses all the task's deadlines.

VI. CONCLUSION AND FUTURE WORK

We have presented CA-CBBA, a new learning-based algorithm for decentralized task allocation in networked agents with communication constraints. Two local features of Bron-Kerbosch or BK feature and the value of message are used to learn a decentralized communication policy that adaptively allocates communication resources across the team to achieve

[2]In this example the order of deadlines was: $T_0 < T_9 < T_{10} < T_4 < T_7 < T_{14} < T_{11} < T_1 < T_6 < T_5 < T_{12} < T_{13} < T_2 < T_8 < T_3$.

efficient message passing. The results show that CA-CBBA improves throughput by ∼15% and the convergence time by ∼10x compared to baseline CBBA in adhoc networks, while the learned policy can be generalized to different team sizes and different task numbers beyond the size of the training set. In addition, we have shown CA-CBBA outperforms the other baselines in time-sensitive applications including search and rescue and task scheduling in most of the scenarios. This work is also a first step towards a general goal of co-designing multiagent algorithms and communication protocol to improve real-life performance of multiagent algorithms such as decentralized task allocation. In future, we would like to extend this framework to a general multiagent algorithm by incorporating feature discovery process as part of the RL training loop.

ACKNOWLEDGMENT

The computational support through Amazon Web Services.
Code: https://github.com/mit-acl/CACBBA

REFERENCES

[1] H.-L. Choi, L. Brunet, and J. P. How, "Consensus-based decentralized auctions for robust task allocation," *IEEE Trans. Robot.*, vol. 25, no. 4, pp. 912–926, Aug. 2009.
[2] K.-S. Kim, H.-Y. Kim, and H.-L. Choi, "Minimizing communications in decentralized greedy task allocation," *J. Aerosp. Inf. Syst.*, vol. 16, no. 8, pp. 340–345, Aug. 2019, doi: 10.2514/1.I010624.
[3] L. Johnson, S. Ponda, H.-L. Choi, and J. How, "Improving the efficiency of a decentralized tasking algorithm for UAV teams with asynchronous communications," in *Proc. AIAA Guid., Navigat., Control Conf.*, Aug. 2010, p. 8421.
[4] O. Karasakal, "Air defense missile-target allocation models for a naval task group," *Comput. Oper. Res.*, vol. 35, no. 6, pp. 1759–1770, 2008.
[5] R. J. Meuth, E. W. Saad, D. C. Wunsch, and J. Vian, "Adaptive task allocation for search area coverage," in *Proc. IEEE Int. Conf. Technol. Practical Robot Appl.*, Nov. 2009, pp. 67–74.
[6] A. Hussein, M. Adel, M. Bakr, O. M. Shehata, and A. Khamis, "Multi-robot task allocation for search and rescue missions," *J. Phys., Conf. Ser.*, vol. 570, no. 5, Dec. 2014, Art. no. 052006.
[7] D. Drenjanac, S. D. K. Tomic, L. Klausner, and E. Kühn, "Harnessing coherence of area decomposition and semantic shared spaces for task allocation in a robotic fleet," *Inf. Process. Agricult.*, vol. 1, no. 1, pp. 23–33, Aug. 2014.
[8] J. W. Streefkerk, M. van Esch-Bussemakers, and M. Neerincx, "Context-aware team task allocation to support mobile police surveillance," in *Proc. Int. Conf. Found. Augmented Cognition*. Berlin, Germany: Springer, 2009, pp. 88–97.
[9] M. R. Garey and D. S. Johnson, *Computers and Intractability*, vol. 29. New York, NY, USA: W.H. Freeman, 2002.
[10] J. K. Lenstra and A. H. G. R. Kan, "Complexity of vehicle routing and scheduling problems," *Networks*, vol. 11, no. 2, pp. 221–227, 1981.
[11] H. Huang and T. Zhuo, "Multi-model cooperative task assignment and path planning of multiple UCAV formation," *Multimedia Tools Appl.*, vol. 78, no. 1, pp. 415–436, Jan. 2019, doi: 10.1007/s11042-017-4956-7.
[12] N. Geng, Z. Chen, Q. A. Nguyen, and D. Gong, "Particle swarm optimization algorithm for the optimization of rescue task allocation with uncertain time constraints," *Complex Intell. Syst.*, vol. 7, no. 2, pp. 873–890, Apr. 2021.
[13] Z. Jia, J. Yu, X. Ai, X. Xu, and D. Yang, "Cooperative multiple task assignment problem with stochastic velocities and time windows for heterogeneous unmanned aerial vehicles using a genetic algorithm," *Aerosp. Sci. Technol.*, vol. 76, pp. 112–125, May 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1270963817301220
[14] E. Edison and T. Shima, "Integrated task assignment and path optimization for cooperating uninhabited aerial vehicles using genetic algorithms," *Comput. Oper. Res.*, vol. 38, no. 1, pp. 340–356, 2011.
[15] G. Xu, T. Long, Z. Wang, and L. Liu, "Target-bundled genetic algorithm for multi-unmanned aerial vehicle cooperative task assignment considering precedence constraints," *Proc. Inst. Mech. Eng., G, J. Aerosp. Eng.*, vol. 234, no. 3, pp. 760–773, Mar. 2020.
[16] G. Oh, Y. Kim, J. Ahn, and H. L. Choi, "Market-based distributed task assignment of multiple unmanned aerial vehicles for cooperative timing mission," *J. Aircr.*, vol. 54, no. 6, pp. 2298–2310, 2017.
[17] M. B. Dias, R. Zlot, N. Kalra, and A. Stentz, "Market-based multi-robot coordination: A survey and analysis," *Proc. IEEE*, vol. 94, no. 7, pp. 1257–1270, Jul. 2006.
[18] L. Hunsberger and B. J. Grosz, "A combinatorial auction for collaborative planning," in *Proc. 4th Int. Conf. MultiAgent Syst.*, 2000, pp. 151–158.
[19] H. A. Kurdi, E. Aloboud, M. Alalwan, S. Alhassan, E. Alotaibi, G. Bautista, and P. J. How, "Autonomous task allocation for multi-UAV systems based on the locust elastic behavior," *Appl. Soft Comput.*, vol. 71, pp. 110–126, Oct. 2018.
[20] A. Alhaqbani, H. Kurdi, and K. Youcef-Toumi, "Fish-inspired task allocation algorithm for multiple unmanned aerial vehicles in search and rescue missions," *Remote Sens.*, vol. 13, no. 1, p. 27, Dec. 2020.
[21] W. Wu, N. Cui, W. Shan, and X. Wang, "Distributed task allocation for multiple heterogeneous UAVs based on consensus algorithm and online cooperative strategy," *Aircr. Eng. Aerosp. Technol.*, vol. 90, no. 9, pp. 1464–1473, Nov. 2018.
[22] E. Carrillo, S. Yeotikar, S. Nayak, M. K. M. Jaffar, S. Azarm, J. W. Herrmann, M. Otte, and H. Xu, "Communication-aware multi-agent metareasoning for decentralized task allocation," *IEEE Access*, vol. 9, pp. 98712–98730, 2021.
[23] M. Rantanen, N. Mastronarde, J. Hudack, and K. Dantu, "Decentralized task allocation in lossy networks: A simulation study," in *Proc. 16th Annu. IEEE Int. Conf. Sens., Commun., Netw. (SECON)*, Jun. 2019, pp. 1–9.
[24] L. B. Johnson, "Decentralized task allocation in communication contested environments," Ph.D. dissertation, Massachusetts Inst. Technol., Cambridge, MA, USA, 2016.
[25] S. Nayak, S. Yeotikar, E. Carrillo, E. Rudnick-Cohen, M. K. M. Jaffar, R. Patel, S. Azarm, J. W. Herrmann, H. Xu, and M. Otte, "Experimental comparison of decentralized task allocation algorithms under imperfect communication," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 572–579, Apr. 2020.
[26] M. Otte, M. J. Kuhlman, and D. Sofge, "Auctions for multi-robot task allocation in communication limited environments," *Auton. Robots*, vol. 44, nos. 3–4, pp. 547–584, Mar. 2020.
[27] K. M. Ormazabal, "The law of diminishing marginal utility in alfred Marshall's principles of economics," *Eur. J. Hist. Econ. Thought*, vol. 2, no. 1, pp. 91–126, Mar. 1995.
[28] L. Bertuccelli, H.-L. Choi, P. Cho, and J. How, "Real-time multi-UAV task assignment in dynamic and uncertain environments," in *Proc. AIAA Guid., Navigat., Control Conf.*, 2009, p. 5776.
[29] L. Johnson, S. Ponda, H.-L. Choi, and J. How, "Asynchronous decentralized task allocation for dynamic environments," in *Proc. Infotech@ Aerospace*, 2011, p. 1441, doi: 10.2514/6.2011-1441.
[30] N. Buckman, H.-L. Choi, and J. P. How, "Partial replanning for decentralized dynamic task allocation," in *Proc. AIAA Scitech Forum*, 2019, p. 0915, doi: 10.2514/6.2019-0915.
[31] D. Di Paola, D. Naso, and B. Turchiano, "Consensus-based robust decentralized task assignment for heterogeneous robot networks," in *Proc. Amer. Control Conf.*, Jun. 2011, pp. 4711–4716.
[32] D.-H. Lee, S. A. Zaheer, and J.-H. Kim, "Ad hoc network-based task allocation with resource-aware cost generation for multirobot systems," *IEEE Trans. Ind. Electron.*, vol. 61, no. 12, pp. 6871–6881, Dec. 2014.
[33] S. K. Mishra, S. Mishra, A. Alsayat, N. Z. Jhanjhi, M. Humayun, K. S. Sahoo, and A. K. Luhach, "Energy-aware task allocation for multi-cloud networks," *IEEE Access*, vol. 8, pp. 178825–178834, 2020.
[34] M. Lujak and A. Fernández, "Distributed multi-robot coordination combining semantics and real-time scheduling," in *Proc. 10th Iberian Conf. Inf. Syst. Technol. (CISTI)*, Jun. 2015, pp. 1–6.
[35] X. Li and X. Zhang, "Multi-task allocation under time constraints in mobile crowdsensing," *IEEE Trans. Mobile Comput.*, vol. 20, no. 4, pp. 1494–1510, Apr. 2021.
[36] S. Amador, S. Okamoto, and R. Zivan, "Dynamic multi-agent task allocation with spatial and temporal constraints," in *Proc. AAAI Conf. Artif. Intell.*, 2014, vol. 28, no. 1, pp. 1–7.
[37] A. Netzer, A. Meisels, and R. Zivan, "Distributed envy minimization for resource allocation," *Auton. Agents Multi-Agent Syst.*, vol. 30, no. 2, pp. 364–402, Mar. 2016.

[38] B. Mu, G. Chowdhary, and J. P. How, "Efficient distributed sensing using adaptive censoring-based inference," *Automatica*, vol. 50, no. 6, pp. 1590–1602, Jun. 2014.
[39] D. Kim, S. Moon, D. Hostallero, W. J. Kang, T. Lee, K. Son, and Y. Yi, "Learning to schedule communication in multi-agent reinforcement learning," 2019, *arXiv:1902.01554*.
[40] H. Mao, Z. Gong, Z. Zhang, Z. Xiao, and Y. Ni, "Learning multi-agent communication under limited-bandwidth restriction for internet packet routing," 2019, *arXiv:1903.05561*.
[41] E. de la Hoz, J. M. Gimenez-Guzman, I. Marsa-Maestre, L. Cruz-Piris, and D. Orden, "A distributed, multi-agent approach to reactive network resilience," in *Proc. 16th Conf. Auton. Agents MultiAgent Syst.*, 2017, pp. 1044–1053.
[42] R. Zivan, H. Yedidsion, S. Okamoto, R. Glinton, and K. Sycara, "Distributed constraint optimization for teams of mobile sensing agents," *Auton. Agents Multi-Agent Syst.*, vol. 29, no. 3, pp. 495–536, May 2015.
[43] A. M. Tabakhi, W. Yeoh, R. Tourani, F. Natividad, and S. Misra, "Communication-sensitive pseudo-tree heuristics for DCOP algorithms," *Int. J. Artif. Intell. Tools*, vol. 27, no. 7, Nov. 2018, Art. no. 1860008.
[44] K.-S. Kim, H.-Y. Kim, and H.-L. Choi, "A bid-based grouping method for communication-efficient decentralized multi-UAV task allocation," *Int. J. Aeronaut. Space Sci.*, vol. 21, no. 1, pp. 290–302, Mar. 2020.
[45] I. Demirkol, C. Ersoy, and F. Alagoz, "MAC protocols for wireless sensor networks: A survey," *IEEE Commun. Mag.*, vol. 44, no. 4, pp. 115–121, Apr. 2006.
[46] P. Huang, L. Xiao, S. Soltani, M. W. Mutka, and N. Xi, "The evolution of mac protocols in wireless sensor networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 1, pp. 101–120, 1st Quart., 2013.
[47] A. De Domenico, E. Strinati, and M.-G. Di Benedetto, "A survey on MAC strategies for cognitive radio networks," *IEEE Commun. Surveys Tuts.*, vol. 14, no. 1, pp. 21–44, 1st Quart., 2012.
[48] K. Xu, M. Gerla, and S. Bae, "How effective is the IEEE 802.11 RTS/CTS handshake in ad hoc networks," in *Proc. Global Telecommun. Conf. (GLOBECOM)*, vol. 1, 2002, pp. 72–76.
[49] Y.-C. Liu, J. Tian, C.-Y. Ma, N. Glaser, C.-W. Kuo, and Z. Kira, "Who2com: Collaborative perception via learnable handshake communication," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, May 2020, pp. 6876–6883.
[50] H. Mao, Z. Zhang, Z. Xiao, Z. Gong, and Y. Ni, "Learning agent communication under limited bandwidth by message pruning," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 4, pp. 5142–5149.
[51] M. E. Morocho-Cayamcela, J. N. Njoku, J. Park, and W. Lim, "Learning to communicate with autoencoders: Rethinking wireless systems with deep learning," in *Proc. Int. Conf. Artif. Intell. Inf. Commun. (ICAIIC)*, Feb. 2020, pp. 308–311.
[52] J. N. Foerster, Y. M. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," in *Proc. NIPS*, 2016, pp. 1–13.
[53] V. V. Kapadia, S. N. Patel, and R. H. Jhaveri, "Comparative study of hidden node problem and solution using different techniques and protocols," 2010, *arXiv:1003.4070*.
[54] T. Moscibroda and R. Wattenhofer, "Efficient computation of maximal independent sets in unstructured multi-hop radio networks," in *Proc. IEEE Int. Conf. Mobile Ad-Hoc Sensor Syst.*, Oct. 2004, pp. 51–59.
[55] C. Bron and J. Kerbosch, "Algorithm 457: Finding all cliques of an undirected graph," *Commun. ACM*, vol. 16, no. 9, pp. 575–577, Sep. 1973.
[56] E. A. Akkoyunlu, "The enumeration of maximal cliques of large graphs," *SIAM J. Comput.*, vol. 2, no. 1, pp. 1–6, Mar. 1973.
[57] N. Saxena, A. Roy, and J. Shin, "Dynamic duty cycle and adaptive contention window based QoS-MAC protocol for wireless multimedia sensor networks," *Comput. Netw.*, vol. 52, no. 13, pp. 2532–2542, Sep. 2008.
[58] Y. Rao, C. Deng, G. Zhao, Y. Qiao, L.-Y. Fu, X. Shao, and R.-C. Wang, "Self-adaptive implicit contention window adjustment mechanism for QoS optimization in wireless sensor networks," *J. Netw. Comput. Appl.*, vol. 109, pp. 36–52, May 2018.
[59] R. Lowe, Y. Wu, A. Tamar, J. Harb, P. Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6382–6393.
[60] C. Boutilier, "Planning, learning and coordination in multiagent decision processes," in *Proc. 6th Conf. Theor. Aspects Rationality Knowl. (TARK)*. San Francisco, CA, USA: Morgan Kaufmann, 1996, pp. 195–210.
[61] D. A. Spielman, "Spectral graph theory and its applications," in *Proc. 48th Annu. IEEE Symp. Found. Comput. Sci. (FOCS)*, Oct. 2007, pp. 29–38.
[62] *Propagation Models*, TETCOS, Bengaluru, Karnataka, 2019.
[63] M. Hatay, "Empirical formula for propagation loss in land mobile radio services," *IEEE Trans. Veh. Technol.*, vol. VT-29, no. 3, pp. 317–325, Aug. 1980.

SHARAN RAJA (Member, IEEE) was a Graduate Student with the Center for Computational Science and Engineering and a Research Assistant with the Aerospace Controls Laboratory, MIT, during the time of this project. His research interests include multi-agent systems, reinforcement learning, and task allocation algorithms.

GOLNAZ HABIBI (Member, IEEE) received the B.Sc. degree in electrical and control engineering from the K. N. Toosi University of Technology, Iran, in 2005, the M.Sc. degree in control engineering from Tarbiat Modares University, Iran, in 2007, and the Ph.D. degree in computer science from Rice University, in 2015. She is currently a Research Scientist with the Department of Aeronautics and Astronautics, MIT. She is broadly interested in robotics, control systems, machine learning, and multi-agent systems. Her current research interests include visual navigation, reliable communication, and the safe and reliable autonomous agents. Her paper has been nominated for the Best Student Paper Award in DARS 2012. She received the K2I Award by Chevron Company in 2013.

JONATHAN P. HOW (Fellow, IEEE) received the B.A.Sc. degree in aerospace from the University of Toronto, in 1987, and the S.M. and Ph.D. degrees in aeronautics and astronautics from the Massachusetts Institute of Technology (MIT), in 1990 and 1993, respectively. He studied for 1.5 years at MIT as a Postdoctoral Associate. Prior to joining MIT, in 2000, he was an Assistant Professor with the Department of Aeronautics and Astronautics, Stanford University. He is currently the Richard C. Maclaurin Professor of aeronautics and astronautics with the MIT. He is the Director of the Ford-MIT Alliance. He was the Planning and Control Lead of the MIT DARPA Urban Challenge Team. His research interests include robust planning and learning under uncertainty with an emphasis on multi-agent systems. He was elected to the Board of Governors of the IEEE Control System Society (CSS), in 2019. From 2014 to 2017, he was a member of the USAF Scientific Advisory Board (SAB). He is a member of the IEEE CSS Technical Committee on Aerospace Control and the Technical Committee on Intelligent Control. He is a fellow of AIAA. His work has been recognized with multiple awards, including the 2020 AIAA Intelligent Systems Award. He was an Area Chair of the International Joint Conference on Artificial Intelligence (2019) and will be the Program Vice-Chair (tutorials) of the Conference on Decision and Control (2021). He was the Editor-in-Chief of the *IEEE Control Systems Magazine* (2015–2019). He is an Associate Editor for the *AIAA Journal of Aerospace Information Systems* and the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS.