Experimental Comparison of Decentralized Task Allocation Algorithms Under Imperfect Communication

Sharan Nayak, Suyash Yeotikar, Estefany Carrillo, Eliot Rudnick-Cohen, Mohamed Khalid M. Jaffar, Ruchir Patel, Shapour Azarm, Jeffrey W. Herrmann, Huan Xu, and Michael Otte

***Abstract*—We compare the performance of five state of the art decentralized task allocation algorithms under imperfect communication conditions. The decentralized algorithms we consider are CBAA, ACBBA, DHBA, HIPC and PI. All algorithms are evaluated using three different models of communication, including the Bernoulli model, the Gilbert-Elliot model, and the Rayleigh Fading model. All 15 of the resulting combinations of an algorithm with a communication model are evaluated in two different problem scenarios: (1) *Collaborative visit*, a scenario in which the agents have to collaboratively visit known stationary targets. (2) *Collaborative search and visit*, a scenario in which the agents have to collaboratively search for and then visit unknown stationary target locations. Each algorithm is evaluated in each scenario using two performance measures: (1) the maximum distance traveled by any agent (2) the maximum number of messages sent by any agent. Real-time experimental simulations show the trade-offs that exists between these five algorithms at different communication conditions.**

***Index Terms*—Distributed robot systems, task planning, networked robots.**

I. Introduction

TEAMS of Unmanned Aerial Vehicles (UAV) have been proposed for use in applications such as search and rescue [1], firefighting [2] and surveillance and reconnaissance [3]. In these applications, the team members or *agents* need to communicate in order to coordinate the assignment of tasks and verify task completion. However, real world wireless communication is often unreliable, degraded, or constrained, due to fading, path loss and interference among other issues [4] which impacts coordination between agents.

The process of assigning tasks to agents in a team is known as task allocation. There are two groups of task allocation algorithms — centralized and decentralized [5], [6]. Centralized algorithms use the notion of a "master" agent who computes and assigns tasks for each agent whereas decentralized algorithms do not have a master and all agents participate in computing and assigning tasks. The performance of both groups of algorithms degrades under imperfect communication. Centralized algorithms are susceptible to packet drops between master and agent which has been studied in [7]. However, little analysis has been done on the performance of *decentralized* task allocation algorithms under imperfect communication (Fig. 1).

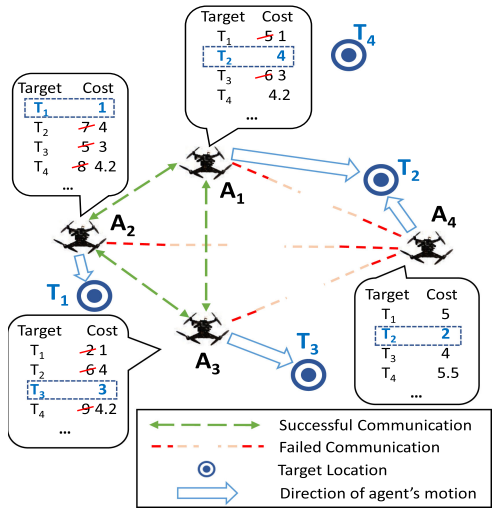

Fig. 1. Agents A_1, A_2, A_3 and A_4 use decentralized task allocation (under imperfect communication) to visit all known targets $T_1, T_2, T_3, T_4 \ldots$. The costs represent the distance to targets. Agents A_1, A_2, A_3 exchange costs (successful communication) and coordinate such that each agent visits a unique target. However, agent A_4 visits same target T_2 as A_1 because of loss of coordination (failed communication).

The main contribution of this letter is a comparison of five decentralized task allocation algorithms (CBAA [8], ACBBA [9], [10], DHBA [11], HIPC [12], [13] and PI [14]) across many communication quality levels using three different communication models: Bernoulli [7], Gilbert-Elliot (G.E.) [15], [16] and Rayleigh Fading [17]. This is useful because it highlights differences in the performance of the decentralized algorithms across different communication conditions. Additional contributions include:

1) This is the first systematic comparison of more than two *decentralized* algorithms.
2) We compare algorithms on two different scenarios, the second of which has not been studied extensively: (i) The *Collaborative visit* scenario: agents collaboratively visit

Manuscript received September 10, 2019; accepted December 16, 2019. Date of publication January 3, 2020; date of current version January 15, 2020. This letter was recommended for publication by Associate Editor Dr. P. Ogren and Editor N. Y. Chong upon evaluation of the reviewers' comments. This work was supported by the Grant FA8750-18-2-0114 from the Air Force Research Laboratory (AFRL), USA. *(Corresponding author: Sharan Nayak.)*

The authors are with the A. James Clark School of Engineering, University of Maryland, College Park, MD 20742 USA (e-mail: snayak18@umd.edu; suyashy@umd.edu; ecarril2@umd.edu; erudnick@umd.edu; khalid26@umd.edu; rpatel18@umd.edu; azarm@umd.edu; jwh2@umd.edu; mumu@umd.edu; otte@umd.edu).

Digital Object Identifier 10.1109/LRA.2019.2963646

known stationary targets. (ii) The *Collaborative search and visit* scenario: agents collaboratively search for unknown stationary targets and then visit them.

The two performance measures considered in our study are the maximum (max) distance traveled by any agent and the max number of messages sent by any agent. All 30 combinations of algorithm, communication model, and scenario are evaluated with respect to both performance measures. Assuming constant velocity, the max distance traveled measure is proportional to the mission completion time, i.e., time taken by agents to complete all tasks.

The rest of the letter is organized as follows. Section II provides a discussion on related work. Section III contains the problem definition for the two scenarios used in the experiments. Section IV describes the decentralized algorithms and communication models. Section V explains the framework used for running the experiments, the design of experiments, and determination of optimal parameters for algorithms. Section VI contains an analysis of experimental data and a discussion of results. Section VII concludes the letter by summarizing our contributions and main results.

II. Related Work

There are a variety of decentralized task allocation algorithms discussed in the existing literature. A majority of these are market based approaches that use auctions. Two widely used auction based approaches are Consensus Based Auction Algorithm (CBAA) and the Consensus Based Bundle Algorithm (CBBA) [8]. CBAA is *single task assignment based* where an agent is assigned a single task at a time. CBBA is *bundle task assignment based* where an agent is assigned multiple tasks.

There have been several improvements made to the CBBA algorithm, most notably, the Asynchronous Consensus Based Bundle Algorithm (ACBBA) [9], [10], Hybrid Information and Plan Consensus (HIPC) algorithm [12], [13], and Performance Impact (PI) Algorithm [14]. ACBBA eliminates the need for coordinated synchronous communication in the consensus phase of CBBA, thereby minimizing the number of messages used while retaining the convergence properties of CBBA. HIPC merges ideas of global situational awareness consensus, global plan consensus, and local plan consensus, and also handles cases where the network conditions and mission objectives are dynamic across the team. PI tries to optimize the mathematical objective for the problem and provides conflict-free solutions [14].

Another class of decentralized algorithms use optimization techniques to solve the task allocation problem. They can be divided into deterministic or stochastic optimization based approaches. The Decentralized Hungarian Based Algorithm (DHBA) [11] is an example of a deterministic optimization algorithm that uses the Hungarian method [18] to perform task allocation. In contrast, Wang *et al.* [19] utilize the stochastic ant-colony optimization algorithm [20] to solve the task allocation problem.

A comprehensive comparison of decentralized task allocation algorithms has yet to appear in the literature. However, several pairwise comparisons between two different algorithms exist, which we now survey. Johnson *et al.* [10] compare ACBBA with the original CBBA and find that ACBBA uses less number of messages both in full connected network and line network topologies. Ismail *et al.* [11] compare DHBA and show that unlike CBAA, DHBA always finds the optimal solution under perfect communication. Johnson *et al.* [13] compare HIPC algorithm with CBBA and show that HIPC outperforms CBBA in terms of the number of messages exchanged, number of conflicts and number of iterations. Zhao *et al.* [14] compare PI algorithm with CBBA in a variety of scenarios and demonstrate that PI outperforms CBBA. While many papers have compared their proposed approaches against CBAA or CBBA, they have not compared against each other. Our work fills an existing gap in the literature by comparing five decentralized task allocation algorithms (CBAA, ACBBA, DHBA, HIPIC and PI) across a variety of communication conditions, problem scenarios, while using three different communication models and two different performance metrics. We choose these algorithms because they are highly cited and representative of a range of decentralized algorithms available in literature. Although most of these algorithms are auction algorithms, they vary in the number of task assignments required, bid generation, and the type of messages exchanged.

Previous works have used different communication models for comparing decentralized algorithms. Ismail. *et al.* [11] use a simple disc communication model where the size of the workspace area is $n \times n$ units and the communication range is $n/2$ to compare DHBA and CBAA. Johnson *et al.* [10] use two different network topologies—a fully connected and line network, to compare ACBBA with CBBA. Zhao *et al.* [14] use different network topologies like row, star, circular and mesh networks to simulate different communication scenarios to test PI. Johnson *et al.* [13] evaluate HIPC by varying degrees of network connectivity. Unlike previous works that simulate varying communication with different network topologies, the current letter assumes a full mesh topology (in which every agent *attempts* to communicate with every other agent) and then messages are dropped according to the Bernoulli, G.E., and Rayleigh Fading models. These models account for path loss, fading, burst errors, and bandwidth saturation inherent in many communication channels.

III. Problem Definition

We formulate the decentralized task allocation problem as follows: Let $\mathcal{A} = \{a_1, \ldots, a_n\}$ be the set of agents and $\mathcal{T} = \{t_1, \ldots, t_m\}$ be the set of tasks. Let $S_i \subseteq \mathcal{T}$ be a sequence of l tasks assigned to agent i and let $q(S_i)$ be its cost. The goal of the decentralized task allocation problem is to have the sequence S_i for all agents i satisfy the condition $\mathcal{T} = \bigcup_{i=1}^{n} S_i$. We define $\mathcal{T}$ differently in each of the two sub problems of decentralized task allocation (Fig. 2) studied in this letter:

A. Collaborative Visit Scenario

In this scenario, the tasks correspond to a set of *a priori* known stationary targets $\mathcal{U} = \{u_1, \ldots, u_n\}$ that the agents have to visit in a map of size $M \times M$. We define $\mathcal{T} \triangleq \mathcal{U}$ and S_i to be sequence of l targets to be visited by agent i. A target is considered to be visited if an agent moves within a threshold distance δd_T of the target's location. The mission is completed when all targets are visited by at least one of the agents.

B. Collaborative Search and Visit Scenario

In this scenario, the tasks corresponds to a set of unknown stationary targets $\mathcal{U} = \{u_1, \ldots, u_n\}$ that the agents have to

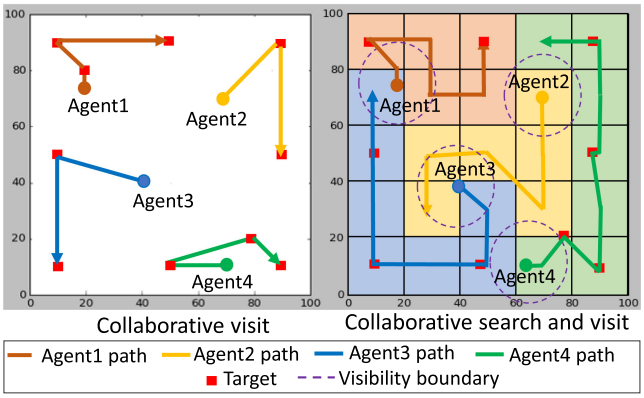

Fig. 2. Two scenarios used to compare decentralized algorithms in imperfect communication conditions.

search and visit in a map of size $M \times M$. To facilitate searching, the search space is divided into a set of user defined grid cells $\mathcal{G} = \{g_1, \ldots, g_r\}$ about which the agents have knowledge *a priori*. As the agents move in the map, they are capable of discovering new targets that lie within their sensor radius R_v. Thus the set $\mathcal{U}$ is empty at the start of the simulation and gets populated as more targets are found. This is in contrast to the visit scenario where the elements of $\mathcal{U}$ are fully known at the start of the simulation and do not change. A cell is said to be searched when an agent reaches the center of the cell, i.e., because the cell is assumed to be completely within the sensor radius of the agent when the agent is located at the center of the cell. The agents share the information about the newly discovered targets with other agents. We define $\mathcal{T} \triangleq \mathcal{U} \cup \mathcal{G}$ and S_i to be the sequence of l tasks (search and visit) in $\mathcal{T}$ that are assigned to agent i. The mission is completed when all cells are searched and all targets are visited by at least one of the agents.

The *min-max* objective $\mathcal{O}(\mathcal{X})$ considered by all agents in both scenarios is to find a task assignment $\mathcal{X}^* = \{S_1 \ldots S_n\}$ such that

$$\mathcal{O}(\mathcal{X}^*) = \min_{\mathcal{X}} \left(\max_{i \in \mathcal{A}} q(S_i) \right).$$

The tasks of searching cells and visiting targets both involve visiting locations. Thus, completing the task set S_i involves visiting l locations. We define the cost function $q(S_i)$ as

$$q(S_i) = C_i + \sum_{k=1}^{l} c_i(t_k)$$

where C_i is the cost accrued by agent i in reaching its current location and $c_i(t_k)$ is agent i's cost of visiting target t_k. In our work, $c_i(t_k)$ is defined as the Euclidean distance from agent i to target t_k. Thus, the min-max objective requires minimizing the maximum distance traveled by any agent.

Assuming constant velocity, the min-max objective corresponds to minimizing the time required by the UAV *team* to visit a set of locations (and time is an important consideration in emergency management and large scale surveillance operations [21]). A second objective that we consider is the min-max number of transmitted messages. Min-max transmissions is a useful metric in scenarios involving limited network connectivity, bandwidth, or stealth.

Algorithm 1: CBAA on agent i.

1: **procedure** CBAA$(d_i, W_i, \mathcal{T}, \mathcal{I})$
2: $t_i \leftarrow None$
3: **for** $k \leftarrow 1$ to $\mathcal{I}$ **do**
4: $(t_i, W_i) \leftarrow Assignment(t_i, d_i, W_i, \mathcal{T})$
5: $SendBids(W_i)$
6: $(t_i, W_i) \leftarrow Consensus(t_i, W_i)$
7: **return** t_i

Algorithm 2: ACBBA on agent i.

1: **procedure** ACBBA$(d_i, W_i, \mathcal{T}, \mathcal{I}, \mathcal{B})$
2: $b_i \leftarrow None$
3: **for** $k \leftarrow 1$ to $\mathcal{I}$ **do**
4: $(b_i, W_i) \leftarrow Assignment(b_i, d_i, W_i, \mathcal{T}, \mathcal{B})$
5: $SendBids(W_i)$
6: $(b_i, W_i) \leftarrow Consensus(b_i, W_i, \mathcal{B})$
7: **return** b_i

Algorithm 3: PI on Agent i.

1: **procedure** PI$(d_i, S_i, \mathcal{T}, \mathcal{I}, \mathcal{B})$
2: $b_i \leftarrow None$
3: **for** $k \leftarrow 1$ to $\mathcal{I}$ **do**
4: $(b_i, S_i) \leftarrow TaskInclusion(b_i, d_i, S_i, \mathcal{T}, \mathcal{B})$
5: $SendSignificanceList(S_i)$
6: $(b_i, S_i) \leftarrow ConsensusAndTaskRem(b_i, S_i, \mathcal{B})$
7: **return** b_i

IV. Preliminaries

In this section, we outline the algorithms and communication models used in our analysis.

A. Decentralized Task Allocation Algorithms

We evaluate and compare five decentralized algorithms. The inputs — target list $\mathcal{T}$, current distance traversed d_i by agent i and iteration count $\mathcal{I}$ are provided to all algorithms.

1) CBAA: This single-task assignment algorithm (Algorithm 1) operates in two phases: the *assignment* phase and the *consensus* phase. The *assignment* phase (line 4) consists of each agent determining local bids for all tasks known to be incomplete and greedily assigning itself the lowest bid task t_i. The agent then updates the winning bids list W_i with the lowest bid task and sends it to all the other agents (line 5). The *consensus* phase (line 6) has each agent receiving the winning bids list from other agents and updating its bids list with the lowest bids. The item is awarded to the agent with the best (lowest) bid.

2) ACBBA: This multi-task assignment algorithm (Algorithm 2) operates in two phases- *assignment* phase and the *consensus* phase. The *assignment* phase (line 4) is used for greedily determining an ordered task list/bundle b_i (up to size $\mathcal{B}$) and updating the winning bids list W_i with the bids of the task list. The winning bids list along with winning time stamps is sent out to all other agents (line 5). The *consensus* phase (line 6) has each agent receiving messages from other agents and updating its internal bid list. We reset the current task list whenever a new target is dynamically added (this is also done for the other bundle assignment algorithms we evaluate).

Algorithm 4: DHBA on Agent i.

1: **procedure** DHBA($d_i, \mathcal{T}, \mathcal{I}$)
2: $t_i \leftarrow None$
3: $C_i \leftarrow InitCostMat(d_i, \mathcal{T})$
4: **for** $k \leftarrow 1$ to $\mathcal{I}$ **do**
5: $t_i \leftarrow Assignment(C_i)$
6: $SendCostMatrix(C_i)$
7: $C_i \leftarrow Update(C_i)$
8: **return** C_i

Algorithm 5: HIPC on Agent i.

1: **procedure** HIPC($d_i, W_i, \mathcal{T}, \mathcal{I}$)
2: $b_i \leftarrow None$
3: **for** $k \leftarrow 1$ to $\mathcal{I}$ **do**
4: $(b_i, W_i) \leftarrow TAA(b_i, d_i, W_i, \mathcal{T})$
5: $SendBids(W_i)$
6: $(b_i, W_i) \leftarrow Consensus(b_i, W_i)$
7: **return** b_i

3) PI: This multi-task assignment heuristic algorithm (Algorithm 3) modifies CBBA to utilize a different bid evaluation called "significance" that tries to measure the contribution of a task to the local cost generated by each agent. PI operates in two phases — the *task inclusion* phase and the *consensus and task removal* phase. The *task inclusion* phase (line 4) is used to calculate marginal significance of all tasks not included in the current task bundle b_i and use it to update the task bundle (up to size $\mathcal{B}$) and significance list S_i. This significance list is then sent to other agents (line 5). The *consensus and task removal* phase (line 6) is used for achieving consensus on the significance values of tasks on each agent and for removing tasks that has been outbid by another agent. We use ACBBA consensus rules in our implementation.

4) DHBA: This single-task assignment algorithm (Algorithm 4) first initializes a cost matrix C_i using the current distance traveled d_i and current cost of visiting known uncompleted targets (line 3). It then operates in two phases: the *assignment* phase and the *update* phase. The *assignment* phase (line 5) consists of each agent running the Hungarian algorithm [18] on C_i to get an uncompleted task t_i. The agent broadcasts C_i to all other agents (line 6). The update phase (line 7) has each agent receiving the cost matrix from other agents and updating C_i with the costs from other agents.

5) HIPC: This multi-task assignment algorithm (Algorithm 5) is built on top of the original CBBA. Instead of greedily generating its own task bundle, HIPC tries to solve the task assignment problem for all agents and uses this assignment to generate bids on tasks. This algorithm has two phases — *task allocation* phase and *consensus* phase. The *task allocation* phase (line 4) consists of each agent running a full Task Allocation Algorithm (TAA) to generate agent's task bundle b_i. We use a variation of the nearest neighbors algorithm [22] modified for min-max objective as our TAA implementation. The process of sending bids (line 5) and the *consensus* phase (line 6) is same as that for ACBBA.

B. Communication Models

We use the following three models to simulate unreliable communication channels (assuming an unreliable communication protocol, e.g., like UDP is used):

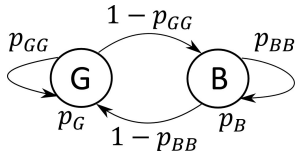

Fig. 3. G.E. model. **G** represents good communication state and **B** represents bad communication state.

TABLE I
COMMUNICATION MODEL COMPARISON

Communication Model	Packet drop	Fading	Distance b/w agents (path loss)	Bandwidth saturation
Bernoulli	✓	×	×	×
G.E.	✓	×	×	✓
Rayleigh Fading	✓	✓	✓	×

1) Bernoulli Model: This model consists of a single parameter p $(0 \leq p \leq 1)$ which is the probability that a message sent by an agent is successfully received by another agent. The communication attempts are independently and identically distributed (i.i.d).

2) Gilbert-Elliot (G.E.) Model: This model is equivalent to a Markov chain that consists of two states, Good and Bad. The two states are used to model different degrees of communication such that the probability of successfully transmitting a message p_G in the good state is greater than in the bad state p_B i.e. $p_G > p_B$. The state transition probability p_{GG} represents the probability of transition from the good state back to good state and p_{BB} represents the probability of transition from bad state to bad state (Fig. 3). The G.E. model can model burst errors and bandwidth saturation [7], [16].

3) Rayleigh Fading Model: Fading refers to the process of variation in the attenuation of a wireless signal due to interference from objects in the environment. These objects cause the wireless signal to be propagated along multiple paths with each path-signal experiencing a different shift in amplitude, frequency and phase and finally interfering constructively or destructively at the receiver. In the Rayleigh fading model, the received channel envelope varies according to the Rayleigh distribution. We use the Inverse Discrete Fourier Transform (IDFT) technique [23] to generate the Rayleigh random variate sequence of size N because of its superlative and efficient variate generation process. We then randomly select a sample from the generated variate sequence to describe the fading power, P_F. Besides fading, we also take into account the attenuation in the transmitted signal due to path loss. The path loss P_{PL} [24] depends on the distance between the transmitting and receiving agent, formulated as

$$P_{PL} = P_{L_0} + 10\gamma \log_{10}\left(\frac{d}{d_0}\right)$$

where d is the distance between the transmitter and receiver, γ is the path loss exponent, P_{L_0} is the path loss at reference distance, d_0. The combined power loss P_L due to fading and path loss is $P_L = P_{PL} + P_F$. The total received power is given by $P_R = P_T - P_L$ where P_T is the transmitted power. P_R is compared with a user-defined sensitivity threshold P_S. A message is dropped if $P_R < P_S$ (Fig. 4).

A comparison of the communication models is provided in Table I.

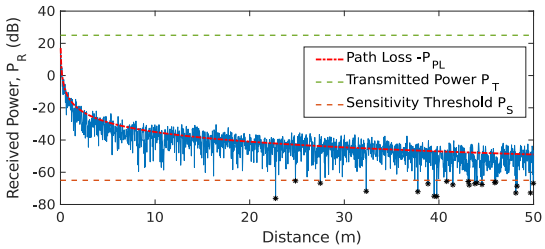

Fig. 4. Figure shows received power P_R of a signal attenuated by Rayleigh fading and path loss. The transmitted power $P_T = 30$ dB and the sensitivity threshold $P_S = -65$ dB. An asterisks '*' represents packets that are dropped.

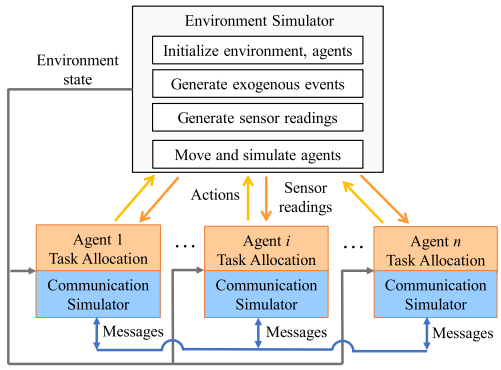

Fig. 5. Framework used for comparing decentralized task allocation algorithms under varying communication.

V. Experimental Setup

This section first describes the simulation framework we use for running experiments, then outlines our experimental design and finally describes our method for determination of optimal parameters for algorithms.

A. Simulation Framework

The simulation framework (Fig. 5) consists of two types of modules: Agent (1 per agent) and Environment (1 total).

Agent: Each agent module represents an independent processing unit that runs the decentralized task allocation algorithms. The algorithms are executed through procedure calls (Section IV) every 0.1 s. Each agent communicates with other agents via a simulation of the communication model. The communication model and its parameters dictate whether an incoming message is accepted or dropped. The algorithms are oblivious to the communication model in use.

Environment Simulator: The environment simulator generates the two dimensional (2D) map for the simulation. The map consists of agents which are modeled as point robots with a constant speed A_s and targets which are modeled as stationary points. The map has a collision free model for physical agent-agent interactions. The environment simulator moves the robots when requested by the agent modules, represented as actions in Fig. 5. It generates odometry sensor readings Fig. 5, which are utilized by the agent modules to determine if an agent has reached a target or not.

The simulation framework is built using the Robot Operating System (ROS) Kinetic framework [25]. The communication simulator is written in C++ and the algorithms and environment simulator are coded in Python.

TABLE II
Parameter Ranges for Instance Generation

Parameter	Range
Number of agents	[5,10]
Number of target clusters (K)	[1,4]
Initial locations of agents	([0, 100],[0, 100])
Cluster centers (μ_i)	([0, 100],[0, 100])
Cluster radii (r)	[5,50]

B. Design of Experiments

We use a randomized design of experiments. An instance is defined as one random sampling of the parameters from the ranges mentioned in Table II. These ranges are universal across all scenarios, communication models and algorithms.

The dimension of the map is $M \times M$, with $M = 100$ units. The target locations in this map are determined by sampling a 2D Gaussian Mixture Model p_{gmm}

$$p_{gmm} = \frac{1}{K}\sum_{i=1}^{K}\mathcal{N}(\mu_i, \Sigma_i)$$

p_{gmm} represents a spatial probability distribution over the map created using multiple Gaussian distributions with the maximum probability at the cluster centers. In this model, the number of clusters K is first fixed. The cluster centers μ_i are set by randomly sampling for numbers in the range [0, 100], and the cluster co-variance Σ_i is set to a diagonal matrix with square of cluster radius as values on the diagonal. The clustering of targets is done to get varying realistic distributions of targets in experiments.

We fix the agent speed $A_s = 6$ units/s and threshold distance $\delta d_T = 0.25$ units for both scenarios. For collaborative search and visit scenario, we fix the number of grid cells $r = 25$ and sensor radius $R_v = 28.28$ units (half length of diagonal of grid cell). For each of the instances, we vary the communication model parameters as follows:

Bernoulli Model: We fix values of p from the range $[0, 1 - \log_{10}(b_i)]$ with b_i varying from 1 to 10 to give a total of 10 communication levels. $p = 1$ represents high communication and $p = 0$ represents low communication.

G.E. Model: We assume the good state to have perfect communication ($p_G = 1$) and the bad state to have no communication ($p_B = 0$). The state transition probabilities p_{GG}, p_{BB} are set to either Low (L), Medium (M) or High (H) with corresponding probabilities of 0.1, 0.5 and 0.9 respectively. The permutation of these 3 values for the tuple (p_{GG}, p_{BB}) gives a total of 9 communication levels. On average, the tuples HL (High, Low) and LH (Low, High) represent high and low communication respectively. We check for state transitions every 1 s.

Rayleigh Fading Model: We fix the model parameters as $N = 64$, chosen as power of 2 for fast computations, $d_0 = 1$ unit, $P_{L_0} = 40$ dB (calculated using the Friis propagation model [26] assuming a 2.4 GHz signal commonly used in many communication networks) and $P_T = 30$ dB. We choose $\gamma = 2.5$ to simulate semi-urban to rural environments (γ ranges between 2 to 6 with 2 for uncluttered space and 6 for densely obstructed urban areas [24]). We vary the sensitivity threshold P_S in the range $[-25, -75]$ dB in -10 dB increments for a total of 6

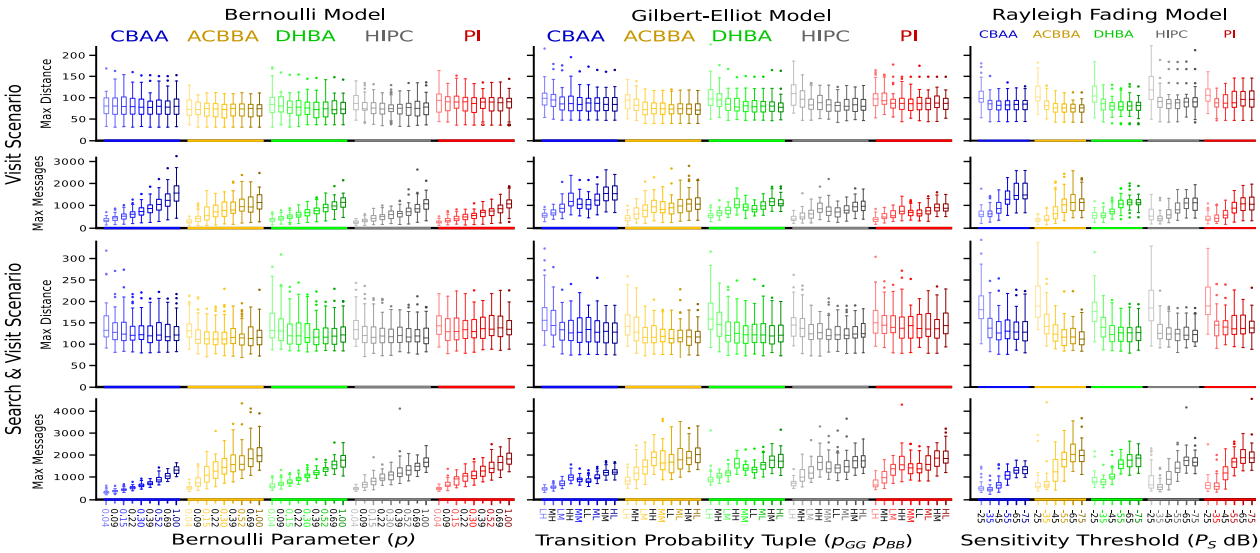

Fig. 6. Box plots for algorithms across different communication models and scenarios. The Bernoulli $p = 0$ case has been omitted in this plot because including the high average distance and message values, for all algorithms at $p = 0$, distorted the plot beyond usefulness.

TABLE III
TOTAL NUMBER OF EXPERIMENTS FOR EACH COMMUNICATION MODEL PER SCENARIO

Com. Model	Com. levels	Instances	Algorithms count	Total experiments
Bernoulli	10	50	5	2500
G.E.	9	50	5	2250
Rayleigh	6	50	5	1500

communication levels. -25 dB and -75 dB represent low and high communication respectively. We note that P_T and range of P_S can be changed by relative offsets without affecting results.

The total number of experiments conducted per communication model per scenario is shown in Table III. The experiments were run on an AMD Ryzen Threadripper 2990WX, 32-core, 3 GHz CPU with 32 GB RAM.

C. Determination of Optimal Parameters of Algorithms

The parameter space of CBAA, DHBA and HIPC contains the max iteration count $\mathcal{I}$, and for ACBBA and PI, it contains both $\mathcal{I}$ and max bundle size $\mathcal{B}$. Values for the $\mathcal{I}$ and $\mathcal{B}$ tuning parameters are chosen as follows: for each $\mathcal{I} \in \{1, 2, 3, 4, 5, 10, 15, 20\}$ and $\mathcal{B} \in \{1, 2, 3, 4, 5, 10, 15, \ldots, 35, 40\}$ (visit scenario) and $\mathcal{B} \in \{1, 2, 3, 4, 5, 10, 15, \ldots, 60, 65\}$ (search and visit scenerio), we run 10 "tuning" experiments with 7 agents and 22 targets *at perfect communication* (7 and 22 represent the median agent and target counts we study across both scenarios). Agent and target starting locations are randomly determined, and then used across all points in the parameter space $\mathcal{I}$, or $(\mathcal{I}, \mathcal{B})$ depending on algorithm, to ensure a fair comparison. We average the results of the 10 experiments separately for each $\mathcal{I}$, $\mathcal{B}$, algorithm, and performance metric. Next, for each algorithm, we select the $\mathcal{I}$ or $(\mathcal{I}, \mathcal{B})$ that yields the lowest min-max distance, on average, and compare its performance to those of the other $\mathcal{I}$ or $(\mathcal{I}, \mathcal{B})$ using the Wilcoxon's signed rank test (WSR) [27] at 5% significance level. If a different $\mathcal{I}$ or $(\mathcal{I}, \mathcal{B})$ with similar distance distribution (according to the WSR test) is found, we choose the value that produces the lowest max message transmission count, on average. The iteration counts $\mathcal{I}$ chosen for CBAA, ACBBA, DHBA, HIPC, PI are $2, 1, 2, 1, 1$, respectively, for visit scenario and $1, 1, 2, 1, 1$ for search and visit scenario, respectively. The max bundle size $\mathcal{B}$ for ABBA, PI are $25, 2$ for the visit scenario and $10, 2$ for the search and visit scenario respectively.

VI. EXPERIMENTAL RESULTS AND DISCUSSION

We visualize the data from our experiments three ways, including: box plots [28] (Fig. 6), Metrics Trade-off plots (MTP)(Fig. 8) and WSR test plots [27] (Fig. 7).

The box plots (Fig. 6) show the mean and distribution of performance data separately for each metric, across all communication models, communication levels, and scenarios. Each box-plot represents 50 instances. We believe that the high interquartile range for min-max distance and messages is due to the randomized agent and target starting locations.

The WSR plots (Fig. 7) show the statistical significance between each pair of algorithms with respect to the two performance metrics. The null hypothesis is that the median difference in the performance metric between two algorithms is zero. There are four colors because the null hypothesis may or may not be rejected for each of the two metrics ($2 \times 2 = 4$).

The MTP plots (Fig. 8) are an alternative data visualization that highlight how the best performing algorithms (at each communication level) have different trade-offs between the two performance metrics. Data points represent the mean performance over 50 instances (for each combination of scenario, communication level, and model), and points at the same communication level are linked with line segments. To declutter the MTP plots, we only draw points for algorithms that have the best performance with respect to some linear combination of the two metrics. Some readers may find the analogy of a Pareto frontier [29] useful in interpreting the MTP plot, in this analogy the markers depict the non-dominated solutions.

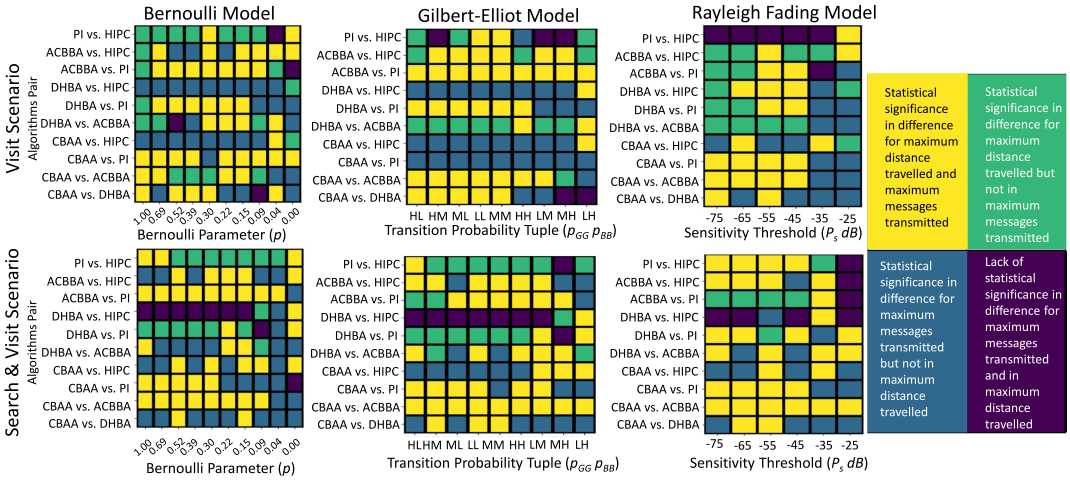

Fig. 7. Wilcoxon tests to determine which algorithms are statistically different across different scenarios and communication models.

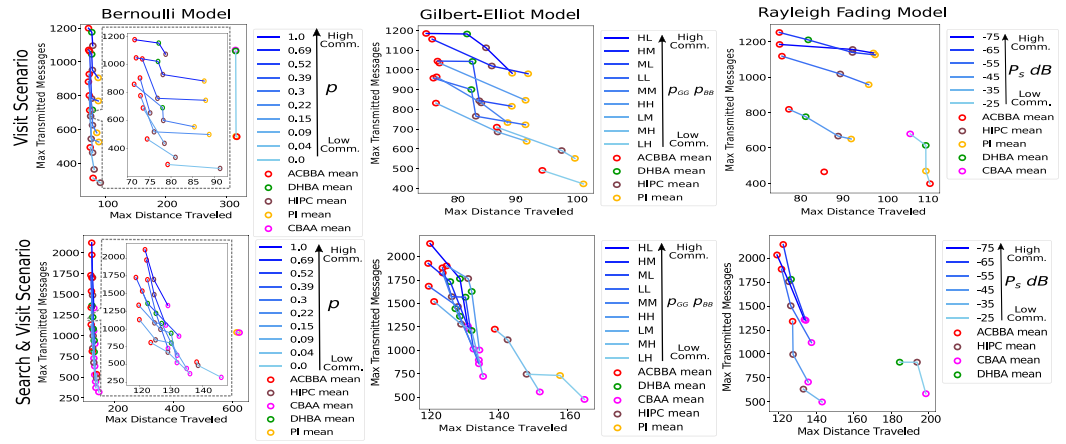

Fig. 8. Metrics Trade-off Plot (MTP) across different scenarios and communication models. The hollow circles represent the algorithms with only the best performing algorithms (non-dominated) shown at each communication level to reduce clutter. The algorithms at the same communication level are joined by same colored line segments for easy visualization. The plot inside the dashed grey rectangle in the Bernoulli plots represents the zoomed version of the data points on the left hand side.

Comparison results for the two scenarios

A. Visit Scenario

At high communication, Figs. 6 and 8 show ACBBA, DHBA, HIPC and PI generally perform better across all communication models. The WSR test for ACBBA vs. DHBA shows statistical difference for the distance travelled metric but not the messages transmitted metric (green squares) for all communication models. In the Bernoulli and Rayleigh models, we observe the same trend for ACBBA vs. HIPC and for ACBBA vs. PI. Since ACBBA performs better on average in terms of min-max distance, we infer that ACBBA performs best at high communication when using the Bernoulli and Rayleigh models. However, there exists a statistical difference for ACBBA vs. PI at high communications (yellow squares) for the G.E. model. Hence a trade-off exists between ACBBA (less distance) and PI (less messages) at high communication.

At low communication, Fig. 6 and Fig. 8 show ACBBA and PI generally perform better across all communication models. ACBBA (and/or PI) perform the best at the zero communication case for the Bernoulli model. At low communication, there exists a trade-off between ACBBA and HIPC and ACBBA and PI when using Bernoulli and G.E. model (yellow squares) respectively. For the lowest communication level we tested on the Rayleigh model, there is only statistically significant differences in messages transmitted for ACBBA vs. CBAA, ACBBA vs. DHBA and ACBBA vs. PI (blue squares). Since ACBBA, sends the lowest number of messages, we conclude that ACBBA performs the best given the lowest communication level tested for the Rayleigh model. Fig. 8 shows that ACBBA also performs best at next higher level (-35 dB) of Rayleigh model.

B. Search and Visit Scenario

At high communication, Fig. 6 and Fig. 8 show ACBBA, HIPC, DHBA and CBAA generally perform better across all communication models. Given the G.E. and Rayleigh models, the WSR test for ACBBA vs. HIPC, ACBBA vs. DHBA and ACBBA vs. CBAA show a statistical difference in both distance travelled and messages transmitted (yellow squares). Also, DHBA vs. HIPC shows no statistical difference (purple squares) and CBAA vs DHBA and HIPC vs CBAA shows statistical

difference in only messages transmitted (blue squares). This implies CBAA outperforms HIPC and DHBA since CBAA transmits less messages. This causes the trade-off to boil down between ACBBA (less distance) and CBAA (less messages) at high communication. The same trade-off conclusion between ACBBA and CBAA can be obtained for the Bernoulli model by using a similar analysis.

At low communication, HIPC and CBAA generally perform the best across all communication models. Given the Bernoulli model, CBAA and PI does best at zero communication. At the next higher communication level, the WSR test for ACBBA vs CBAA shows statistical difference in only messages transmitted. Since CBAA send lower number of messages, we can conclude that CBAA does better than ACBBA. For the G.E. model there is a statistical difference in both performance metrics between HIPC and CBAA. A trade-off exists in selecting HIPC (less distance) and CBAA (less messages). For the Rayleigh model, at -35 dB, we find a trade-off exists in choosing HIPC (less distance) or CBAA (less messages) since there is significant difference in both metrics. However at the lowest communication (-25 dB), CBAA performs best as it sends lower number of messages and there is no significant difference in distance traveled for DHBA vs CBAA.

VII. Conclusion

We compare the performance of five decentralized task allocation algorithms (CBAA, ACBBA, DHBA, HIPC and PI) under imperfect communication. We model imperfect communication using Bernoulli model, Gilbert-Elliot model and Rayleigh Fading Model. We consider two scenarios in our experiments: (1) *Collaborative visit* scenario where the agents collaboratively visit *a priori* known targets (2) *Collaborative search and visit* scenario where the agents collaboratively search for and then visit unknown targets. We evaluate performance using two metrics: the max distance traveled and max number of messages sent by any agent.

The results of our experiments suggest that for the collaborative visit scenario, ACBBA generally performs better than other algorithms at high communication levels given either Bernoulli or Rayleigh models. However, a trade-off with PI (less messages) when using the Gilbert-Elliot model. For the Rayleigh model with low communication, ACBBA performs the best. While for the Bernoulli and Gilbert-Elliot models, ACBBA (less distance) shows a trade-off with HIPC and PI (less messages). For the collaborative search and visit scenario, we find a trade-off exists between ACBBA (less distance) and CBAA (less messages) at high communication levels. At low communication levels, CBAA is generally more desirable, although there is a trade-off with HIPC (in general, if the Gilbert-Elliot model is used; and for the $P_s = -35$ dB level of Rayleigh model).

Possible future directions of our research include: using different parameter sets, objective functions, and new scenarios (such as moving and dynamically added targets).

References

[1] S. Waharte and N. Trigoni, "Supporting search and rescue operations with uavs," in *Proc. Int. Conf. Emerg. Secur. Technologies.*, 2010, pp. 142–147.

[2] J. A. Shaffer, E. Carrillo, and H. Xu, "Hierarchal application of receding horizon synthesis and dynamic allocation for uavs fighting fires," *IEEE Access*, vol. 6, pp. 78 868–78 880, 2018.

[3] N. Nigam, S. Bieniawski, I. Kroo, and J. Vian, "Control of multiple uavs for persistent surveillance: algorithm and flight test results," *IEEE Trans. Control Syst. Technol.*, vol. 20, no. 5, pp. 1236–1251, Sep. 2012.

[4] A. A. Khuwaja, Y. Chen, N. Zhao, M.-S. Alouini, and P. Dobbins, "A survey of channel modeling for uav communications," *IEEE Commun. Surv. Tut.*, vol. 20, no. 4, pp. 2804–2821, Oct.–Dec. 2018.

[5] X. Jia and M. Q.-H. Meng, "A survey and analysis of task allocation algorithms in multi-robot systems," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, 2013, pp. 2280–2285.

[6] A. Khamis, A. Hussein, and A. Elmogy, "Multi-robot task allocation: A review of the state-of-the-art," in *Proc. Cooperative Robots Sensor Netw.* Springer, 2015, pp. 31–51.

[7] M. Otte, M. J. Kuhlman, and D. Sofge, "Auctions for multi-robot task allocation in communication limited environments," *Auton. Robots*, pp. 1–38, 2019.

[8] 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.

[9] L. Johnson, S. Ponda, H.-L. Choi, and J. P. How, "Asynchronous decentralized task allocation for dynamic environments," in *Proc. Infotech@ Aerosp.*, 2011.

[10] 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., Navigation, Control Conf.*, 2010.

[11] S. Ismail and L. Sun, "Decentralized hungarian-based approach for fast and scalable task allocation," in *Proc. IEEE Int. Conf. Unmanned Aircr. Syst.*, 2017, pp. 23–28.

[12] L. Johnson, H.-L. Choi, and J. P. How, "The hybrid information and plan consensus algorithm with imperfect situational awareness," in *Proc. Distrib. Auton. Robot. Syst*. Springer, 2016, pp. 221–233.

[13] L. B. Johnson, H.-L. Choi, and J. P. How, "Hybrid information and plan consensus in distributed task allocation," in *Proc. AIAA Guid., Navigation, ControlConf.*, 2013.

[14] W. Zhao, Q. Meng, and P. W. Chung, "A heuristic distributed task allocation method for multivehicle multitask problems and its application to search and rescue scenario," *IEEE Trans. Cybern.*, vol. 46, no. 4, pp. 902–915, Apr. 2016.

[15] E. N. Gilbert, "Capacity of a burst-noise channel," *Bell Syst. Tech. J.*, vol. 39, no. 5, pp. 1253–1265, 1960.

[16] E. O. Elliott, "Estimates of error rates for codes on burst-noise channels," *Bell Syst. Tech. J.*, vol. 42, no. 5, pp. 1977–1997, 1963.

[17] Y. R. Zheng and C. Xiao, "Simulation models with correct statistical properties for rayleigh fading channels," *IEEE Trans. Commun.*, vol. 51, no. 6, pp. 920–928, Jun. 2003.

[18] H. W. Kuhn, "The hungarian method for the assignment problem," *Nav. Res. Logistics Quart.*, vol. 2, no. 1-2, pp. 83–97, 1955.

[19] J. Wang, Y. Gu, and X. Li, "Multi-robot task allocation based on ant colony algorithm," *J. Comput.*, vol. 7, no. 9, pp. 2160–2167, 2012.

[20] M. Dorigo and M. Birattari, *Ant Colony Optimization*. Springer, 2010.

[21] K. S. V. Narasimha and M. Kumar, "Ant colony optimization technique to solve the min-max single depot vehicle routing problem," in *Proc IEEE. Amer. Control Conf.*, 2011, pp. 3257–3262.

[22] M. M. Solomon, "Algorithms for the vehicle routing and scheduling problems with time window constraints," *Oper. Res.*, vol. 35, no. 2, pp. 254–265, 1987.

[23] D. J. Young and N. C. Beaulieu, "The generation of correlated rayleigh random variates by inverse discrete fourier transform," *IEEE Trans. Commun.*, vol. 48, no. 7, pp. 1114–1127, Jul. 2000.

[24] T. S. Rappaport *et al.*, *Wireless Communications: Principles and Practice*. New Jersey: Prentice hall PTR, 1996, vol. 2.

[25] A. Koubâa, *Robot Operating System (ROS)*. New York; Berlin, Germany: Springer, 2017.

[26] R. Khattak, A. Chaltseva, L. Riliskis, U. Bodin, and E. Osipov, "Comparison of wireless network simulators with multihop wireless network testbed in corridor environment," in *Proc. Int. Conf. Wired/Wireless Internet Commun.* Springer, 2011, pp. 80–91.

[27] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bull.*, vol. 1, no. 6, pp. 80–83, 1945.

[28] D. Ruppert, *Statistics and Data Analysis for Financial Engineering*. New York; Berlin, Germany: Springer, 2011, vol. 13.

[29] D. Ščap, M. Hoić, and A. Jokić, "Determination of the pareto frontier for multiobjective optimization problem," *Trans. FAMENA*, vol. 37, no. 2, pp. 15–28, 2013.