Ensuring Network Connectivity for Decentralized Planning in Dynamic Environments

Ensuring Network Connectivity for Decentralized Planning in Dynamic Environments

Sameera S. Ponda*, Luke B. Johnson†, Han-Lim Choi‡, and Jonathan P. How§

This work addresses the issue of network connectivity for a team of heterogeneous agents operating in a dynamic environment. The Consensus-Based Bundle Algorithm (CBBA), a distributed task allocation framework previously developed by the authors and their colleagues, is introduced as a methodology for complex mission planning, and extensions are proposed to address limited communication environments. In particular, CBBA with Relays leverages information available through already existing consensus phases to predict the network topology at select times and creates relay tasks to strengthen the connectivity of the network. By employing underutilized resources, the presented approach improves network connectivity without limiting the scope of the active agents, thus improving mission performance.

I. Introduction

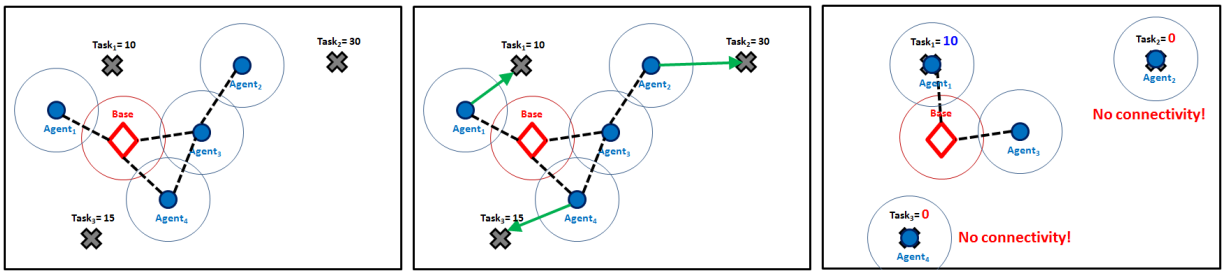
Teams of networked unmanned agents, with heterogeneous capabilities, are regularly employed in autonomous missions including intelligence, surveillance and reconnaissance operations.[1] Mission planning for such teams consists of coordinating the behavior of the agents in order to perform the set of mission tasks as efficiently as possible. Given the heterogeneous nature of the team, some agents are better suited to handle certain tasks than others, leading to different roles and responsibilities within the mission. For example, UAVs equipped with video can be used to perform search and surveillance, human operators can be used for classification tasks, ground teams can be deployed to perform rescue operations, etc. Ensuring proper coordination between the agents in the team is crucial to efficient mission execution, motivating the development of task allocation and planning methods to improve mission coordination. Planning for such teams involves solving significantly complex combinatorial decision problems with many constraints. Tasks may have different locations and time-windows of validity or could require coordinated execution between several agents.[2–6] Agents may have resource limitations, varying capabilities, and specific agent-task compatibility requirements.

A challenging issue that further complicates this problem is that ISR missions are often performed in communication limited environments. Unmanned agent communication systems typically have a limited range or communication radius and/or potential line-of-sight requirements. As a result, as agents move around the environment the network topology is usually dynamic, with varying communication links between the agents. Dynamic communication constraints and network disconnects can cause several issues. For example, performing consensus on information or

*Dept. of Aeronautics and Astronautics, MIT, Cambridge, MA, sponda@mit.edu
†Dept. of Aeronautics and Astronautics, MIT, Cambridge, MA, lbj16@mit.edu
‡Div. of Aerospace Engineering, KAIST, Daejeon, Korea, hanlimc@kaist.ac.kr
§R. C. Maclaurin Professor of Aeronautics and Astronautics, MIT and Associate Fellow of AIAA jhow@mit.edu

situational awareness between agents becomes more complex,[7,8] if employing a distributed planning strategy assignment deconfliction or consensus could be impacted,[6] and, in scenarios where the success of task execution is dependent upon maintaining continuous communication, dynamic network topologies and network disconnects can severely limit the performance of the team.

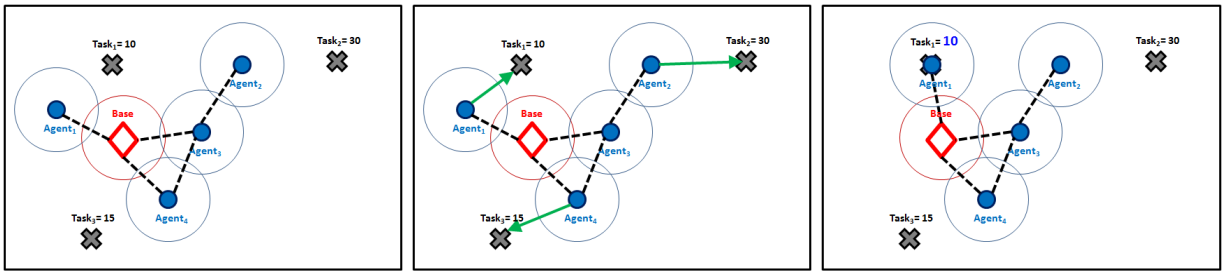
To further illustrate this problem, consider the following motivating example, shown in Figure 1. The depicted mission involves agents performing surveillance around a base station. Unmanned vehicles equipped with video cameras are tasked to travel to select locations and stream video data back to a base station. However, the agents have a limited communication radius, and if they are disconnected from the base at the time of their task execution, then they are unable to stream the surveillance data back, making their efforts fruitless. Figure 1(a) shows the initial agent configuration around the base station, along with the desired task locations and associated values. The halos around each agent and around the base station depict circles of half of the communication radius, therefore, if two circles are intersecting, that implies that the agents are connected (also shown with a dotted black connection line). Figure 1(b) shows the proposed task assignment for the agents. This initial assignment is made without regard to the communication constraints, and as a result, the execution of the mission causes communication disconnects. Figure 1(c) shows the disconnected network, where Agents 2 and 4 receive a score of 0 for their tasks since they are unable to stream back the surveillance data.



(a) Initial mission scenario (b) Proposed assignment (c) Resulting plan execution

Figure 1. Baseline mission planning scenario illustrating the drawbacks of not including limited connectivity constraints in the planning process

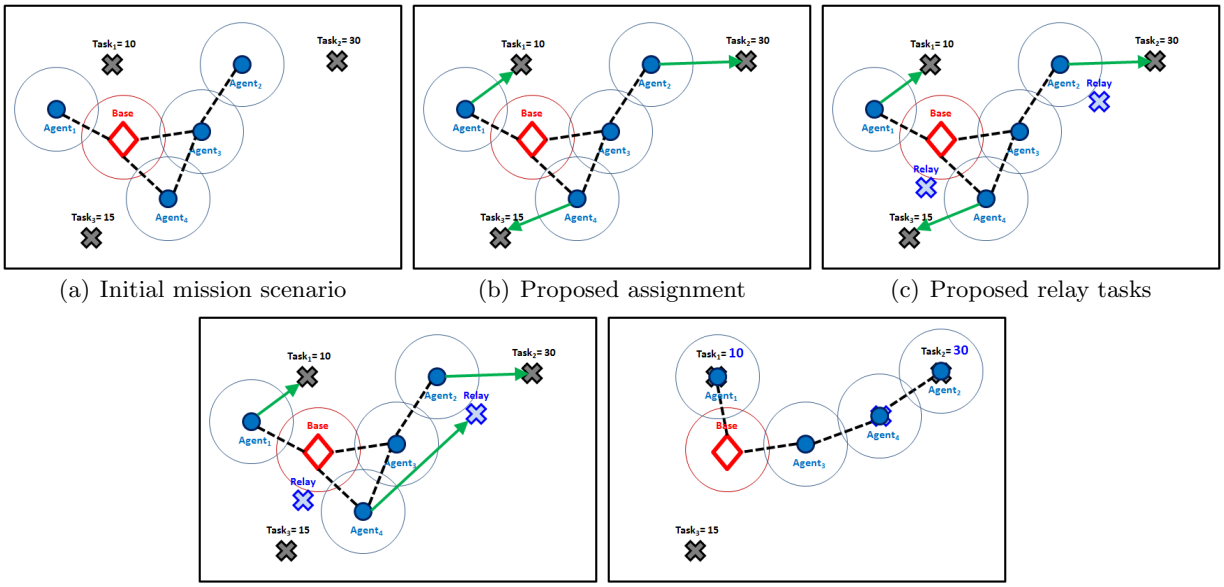
One approach to preventing these network disconnects is to explicitly consider the communication constraints in the planning process. This is typically a very difficult endeavor, which involves the computationally-intensive task of predicting the network topology at every time step. Furthermore, the predicted network topology is highly dependent on the intermediate planning solution, and changing the plan usually requires recomputing the network prediction. This causes loops in the planning process making algorithm convergence non-trivial. Figure 2 illustrates the result of a conservative planning approach, where agents predict the network topology for the proposed assignment, and drop the tasks that will cause network disconnects. Here Figures 2(a) and 2(b) depict the mission scenario and proposed initial plan as before, however in this case Agents 2 and 4 detect that their actions will cause disconnects and conservatively drop their assignments. Figure 2(c) shows the resulting behavior, where only one agent accomplishes a task. This approach guarantees that the network will remain connected but is typically too conservative, since agents can only accomplish tasks in their local vicinity. This is especially detrimental for surveillance missions, where farther tasks usually have higher value, since less information is known farther away from the base and more intelligence is desirable.



(a) Initial mission scenario (b) Proposed assignment (c) Conservative plan execution

Figure 2. Conservative mission planning scenario where agents predict network disconnects prior to plan execution

A better solution to addressing these limited-connectivity issues is to use free agents as communication relays, where data can be transmitted back to the base station via neighboring agents. Consider the case illustrated in Figure 3. Here the agents receive the task locations and make their initial plans as before (Figures 3(a) and 3(b)), predict the network, and detect the potential disconnects, but instead of conservatively dropping their assignments, they create relay tasks that will enable them to maintain connectivity to the base station (Figure 3(c)). At this stage the agents can reevaluate their decisions in light of the new relay task information and determine what is best for the mission. In this scenario, Agent 4 drops its assignment in favor of satisfying the relay task proposed by Agent 2 (Figure 3(d)). The end result is that the team is able to successfully accomplish two of the tasks, leading to a higher mission score (Figure 3(e)). This type of coordinated team behavior typically results in higher mission performance, however, developing planning algorithms that can accomplish this level of coordination is a non-trivial endeavor and remains a key technical challenge.



(a) Initial mission scenario (b) Proposed assignment (c) Proposed relay tasks

(d) Assignment revision (e) Resulting plan execution

Figure 3. Mission planning scenario illustrating the benefits of using relay tasks to prevent network disconnects

Recent work has explored maintaining network connectivity through the use of relays,[9–13] and placing vehicles to maintain communication links.[14] Several of these approaches illustrate the many complications associated with relay planning, however, most of them involve solving complex algorithms that scale poorly as the problem size increases, limiting their real-time applicability for realistic mission scenarios. This paper presents a real-time distributed task allocation framework

for a team of heterogeneous agents performing in a dynamic environment, where varying network topologies and dynamic communication constraints are explicitly considered. Two approaches are presented: one involving predicting the network topology and conservatively dropping tasks that will cause disconnects, and the other exploring the use of relay tasks to strengthen the connectivity of the network. Both approaches have polynomial-time computational complexity and exhibit good scalability for increasing problem sizes. Simulation and experimental results are presented that compare these new algorithms against a baseline distributed planning algorithm, validating the proposed approaches and demonstrating their real-time applicability. These results show that by employing underutilized agents as communication relays, the mission performance can be greatly enhanced.

The layout of the paper is described as follows. Section II presents the task allocation problem formulation, associated challenges and a baseline solution methodology that is scalable to increasing numbers of agents and tasks. Section III describes the proposed extensions to the algorithm to explicitly handle limited communication constraints. Section IV compares the different approaches for an example scenario and presents results that validate the proposed methodology. Finally, Section V presents a summary of the contributions of this work.

II. Problem formulation

Given a heterogeneous team of N_a agents, and a list of N_t tasks with time-windows of validity, the goal of the task allocation algorithm is to find a conflict-free matching of tasks to agents that maximizes some global reward. An assignment is said to be free of conflicts if each task is assigned to no more than one agent. This task assignment problem can be written as the following integer (possibly nonlinear) program, with binary decision variables x_{ij} that are used to indicate whether or not task j is assigned to agent i:

$$\begin{aligned} \max \quad & \sum_{i=1}^{N_a} \left(\sum_{j=1}^{N_t} c_{ij}(\mathbf{p}_i(\mathbf{x}_i, \boldsymbol{\tau}_i)) x_{ij} \right) \\ \text{subject to:} \quad & \sum_{j=1}^{N_t} x_{ij} \leq L_t, \ \forall i \in \mathcal{I} \\ & \sum_{i=1}^{N_a} x_{ij} \leq 1, \ \forall j \in \mathcal{J} \\ & x_{ij} \in \{0,1\}, \ \tau_{ij} \in \{\mathbb{R}^+ \cup \emptyset\}, \quad \forall (i,j) \in \mathcal{I} \times \mathcal{J} \end{aligned} \tag{1}$$

where $x_{ij} = 1$ if agent i is assigned to task j, and $\mathbf{x}_i \triangleq \{x_{i1}, \ldots, x_{iN_t}\}$ is a vector of assignments for agent i, whose j^{th} element is x_{ij}. The index sets for i and j are defined as $\mathcal{I} \triangleq \{1, \ldots, N_a\}$ and $\mathcal{J} \triangleq \{1, \ldots, N_t\}$, representing the index sets for the agents and tasks respectively. The variable length vector $\mathbf{p}_i \triangleq \{p_{i1}, \ldots, p_{i|\mathbf{p}_i|}\}$ represents the path for agent i, an ordered sequence of tasks where the elements are the task indices, $p_{in} \in \mathcal{J}$ for $n = 1, \ldots, |\mathbf{p}_i|$, i.e. its n^{th} element is $j \in \mathcal{J}$ if agent i conducts task j at the n^{th} point along the path. The vector of times, $\boldsymbol{\tau}_i \triangleq \{\tau_{i1}, \ldots, \tau_{i|\mathbf{p}_i|}\}$, denotes the times that agent i proposes to execute the tasks in the path. Each agent can be assigned a maximum of L_t tasks due to limited resource constraints, and the maximum overall number of assignments achievable is given by $N_{\max} \triangleq \min\{N_t, N_a L_t\}$. The current length of the path at each step is denoted by $|\mathbf{p}_i|$ and may be no longer than L_t. The global objective function is assumed to be a sum of local reward values for each agent i, while each local reward is determined as a function of the tasks assigned to the particular agent.

Key assumptions underlying the above problem formulation are:

1. The score c_{ij} that agent i obtains by performing task j is defined as a function of the arrival time τ_{ij} at which the agent executes the task.

2. The optimal arrival time τ_{ij} is a function of the path $\mathbf{p}_i$ that agent i takes.

3. The path $\mathbf{p}_i$ is *uniquely* defined by the assignment vector of agent i, $\mathbf{x}_i$.

The autonomous task allocation process is depicted in Figure 4, where the inputs to the task allocation planner include the task list, typically defined and broadcast by a Mission Control Center, the current network configuration, and agent models for planning predictions. The planning process then allocates the tasks among the team, producing schedules for each of the agents. As the agents interact with the "World" to execute the tasks, the network configuration, agent models and task list are updated to reflect changes in the environment. The planner must replan accordingly to leverage the new information, thus motivating planning algorithms that are scalable and executable in real-time, to handle dynamic environments and communication constraints.

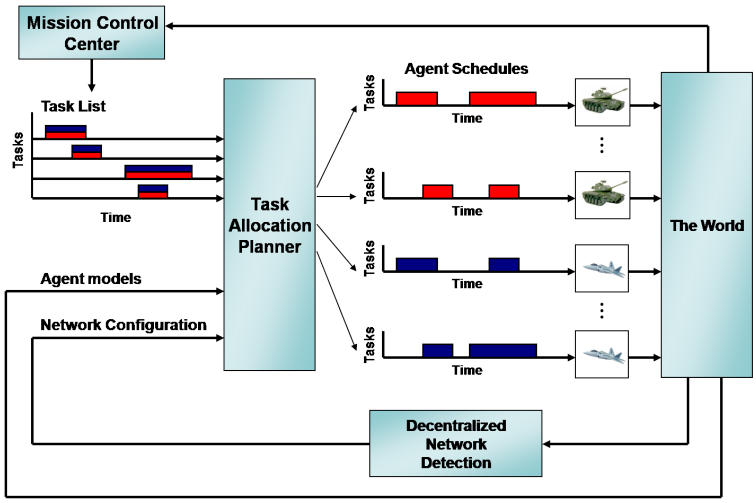


Figure 4. Real-time task allocation architecture for a heterogeneous team

II.A. Consensus-Based Bundle Algorithm (CBBA)

The scoring function in Equation (1) depends on the assignment vectors $\mathbf{x}_i$, on the paths $\mathbf{p}_i$, and on the task arrival times $\boldsymbol{\tau}_i$, for all agents $i \in \mathcal{I}$, which makes this integer programming problem very difficult to solve (NP-hard) for $L_t > 1$ due to all the inherent inter-dependencies.[2] Under these complex constraints, centralized planning methods quickly become infeasible favoring the development of distributed architectures (see [15] and references therein). In addition to alleviating the computational burden by parallelizing the planning process, distributed controllers often act on local information, making response times to changes in situational awareness significantly faster than those achievable under a purely centralized planner. As a result, distributed planning methods which eliminate the need for a central server have been explored.[16–19] Many of these methods often assume perfect communication links with infinite bandwidth, to ensure that agents have the same situational awareness before planning. In the presence of inconsistencies in situational awareness, these distributed tasking algorithms can be augmented with consensus algorithms[15,20–29] to converge on a consistent state before performing the task allocation. Although consensus algorithms guarantee convergence on information, they may take a significant amount of time and often require transmitting large amounts of data.[30]

Other popular distributed task allocation methods involve using auction algorithms,[31,32] which have been shown to efficiently produce sub-optimal solutions. One such algorithm is the Consensus-Based Bundle Algorithm (CBBA),[15] a distributed auction protocol that provides provably good approximate solutions for multi-agent multi-task allocation problems over random network structures. CBBA consists of iterations between two phases: a bundle building phase where each agent greedily generates an ordered bundle of tasks, and a consensus phase where conflicting assignments are identified and resolved through local communication between neighboring agents. CBBA is guaranteed to converge to a conflict-free solution despite possible inconsistencies in situational awareness, and to achieve at least 50% optimality,[15] although empirically its performance is shown to be above 90% optimality.[33] The bidding process runs in polynomial time, demonstrating good scalability with increasing numbers of agents and tasks, making it well suited to real-time dynamic environments. The real-time implementation of CBBA has been demonstrated for heterogeneous teams and the algorithm has been extended to account for timing considerations associated with task execution.[5,6,34,35]

In order to execute the CBBA planning algorithm, the agents must communicate with each other in real-time through established communication links. For the sake of notational simplicity, it is assumed that the state of these links depends only on the distance $d_{ij}(t)$ between two agents i and j, where if this distance is greater than some fixed *communication radius* (R_{COMM}) the connection is broken. Specifically, the network topology can be described by an undirected graph, $G = (V, E)$,[8] where the agents are represented by nodes, V, and the communications links by edges, E. We assume that the communication radius, R_{COMM}, is the same for all agents, implying that the graph, G, is undirected and symmetric. The elements of the graph are given by,

$$G_{ij} = \begin{cases} 1 & \text{if } d_{ij} < R_{COMM} \text{ and } i \neq j \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

As agents move throughout the theater of operation, these communication links are dynamically created and destroyed, leading to varying network topologies. As long as the network remains connected, i.e. there exists some communication path between every pair of agents, either directly or through neighboring agents, the CBBA planning algorithm will produce a conflict-free solution. However, if the network is disconnected, agents are not able to communicate their plans to each other and conflicts may occur. Conflicting assignments negatively impact the overall mission performance, wasting fuel and time by assigning multiple agents to the same tasks.[6] Furthermore, certain tasks rely on maintaining a high bandwidth connection between agents for efficient data transmission. For example, as described in the introduction, surveillance tasks may require agents to stream video back to a base station for processing, which can only be accomplished if the agents maintain connectivity to the base station while the surveillance task is in progress. The next section describes two different approaches that extend the CBBA framework to handle limited connectivity constraints.

III. Improving Connectivity for Dynamic Networks

The research objective is to ensure that the network graph remains connected while agents are executing tasks. An effective way to adjust the network connectivity in real-time is by maintaining or creating links by using other agents as relays. Integrating this notion into the task allocation framework involves predicting the network structure based on task location information and current agents' plans, and creating relay tasks in positions which will strengthen the network connectivity. A key observation in this work is that the network structure at future times can be predicted using task locations and agents' expected path information, which are already available from the

consensus phase in CBBA in the form of winning agent, winning bid, and winning time lists. This information can be leveraged to determine if an assigned task will cause the network to become disconnected at its execution time. Appropriate relay tasks can then be created to connect the network during task execution. In several situations, it is common to have underutilized agents that are "free" for certain periods of time while they await the start time of their next task. These agents can be potentially used to satisfy relay tasks in the network. Furthermore, it is possible to have field agents whose specific purpose is to act as relays between more specialized action agents. The next section describes an extension to CBBA to integrate these relay tasks into the task allocation framework.

III.A. CBBA with Relays: Algorithm Description

The CBBA with Relays algorithm is presented in Algorithm 1 and is described as follows. To initialize the algorithm, the assignment vectors $\mathcal{A}$, the relay list $\mathcal{R}$ and the agents' forbidden task lists $\mathcal{J}_{NA}$ are set to empty. An assignment is obtained using CBBA. Any unassigned relays are deleted along with all their task dependencies (Algorithm 2). This step does not affect the first iteration since there are no relays. The CREATE-RELAYS algorithm (Algorithm 3) then checks the assignment, and for each assigned task, predicts the network structure at the task execution time. If execution of this task will cause the corresponding agent (and possibly others) to become disconnected from the main network, then a group of relay tasks are created to reconnect the agent's sub-network to the main network. The algorithm then iterates by re-running CBBA holding all previous assignments fixed but introducing the relay tasks into the task pool. During the CBBA task selection process, if agents can accommodate relay tasks into their plan, they place appropriate bids on these tasks. A block diagram illustrating the CBBA with Relays algorithm is provided in Figure 5(a).

Algorithm 1 CBBA-RELAYS($\mathcal{I}$, $\mathcal{J}$)

$\mathcal{R} \leftarrow \emptyset$; $\mathcal{A} \leftarrow \emptyset$; $\mathcal{J}_{NA}(i) \leftarrow \emptyset, \forall i \in \mathcal{I}$
converged $\leftarrow$ false
while $\neg$ *converged* **do**
  $\mathcal{A}' \leftarrow$ CBBA($\mathcal{I}$, $\mathcal{J}$, $\mathcal{R}$, $\mathcal{A}$, $\mathcal{J}_{NA}$) (See [15,35] for details)
  for $(r \in \mathcal{R})$ & $(r \notin \mathcal{A}')$ **do**
    $(\mathcal{J}, \mathcal{R}, \mathcal{A}', \mathcal{J}_{NA}) \leftarrow$ DELETE-RELAY(r, $\mathcal{J}$, $\mathcal{R}$, $\mathcal{A}'$, $\mathcal{J}_{NA}$)
  end for
  $(\mathcal{J}, \mathcal{R}', \mathcal{A}', \mathcal{J}_{NA}) \leftarrow$ CREATE-RELAYS($\mathcal{I}$, $\mathcal{J}$, $\mathcal{R}$, $\mathcal{A}'$, $\mathcal{J}_{NA}$)
  if $(\mathcal{R}' = \mathcal{R})$ & $(\mathcal{A}' = \mathcal{A})$ **then**
    converged $\leftarrow$ true
  else
    $\mathcal{R} \leftarrow \mathcal{R}'$; $\mathcal{A} \leftarrow \mathcal{A}'$
  end if
end while
return $\mathcal{A}$

A few intricacies of the algorithm are described below:

- The number of relays required for each disconnected task is a function of the communication radius R_{COMM} and the closest distance between the base sub-network and the disconnected sub-network. The new relays are positioned between the two closest agents in the corresponding sub-networks, the relays scores are the sum of the tasks they are connecting divided by the number of new relays in the group, and their time of validity is equivalent to the maximum

Algorithm 2 DELETE-RELAY(r, $\mathcal{J}$, $\mathcal{R}$, $\mathcal{A}$, $\mathcal{J}_{NA}$)

$\mathcal{R} \leftarrow \mathcal{R} \setminus \{r\}$
for $r' \in$ Grouped-Relays(r) **do**
  $\mathcal{R} \leftarrow \mathcal{R} \setminus \{r'\}$
  $\mathcal{A} \leftarrow \mathcal{A} \setminus \{r'\}$ **if** $r' \in \mathcal{A}$
end for
for $j \in$ Dependent-Tasks(r) **do**
  $keep \sim$ Bernoulli(Normalized-Score(j))
  if $\neg keep$ **then**
    $\mathcal{A} \leftarrow \mathcal{A} \setminus \{j\}$
    $\mathcal{J}_{NA}$(Winning-Agent(j)) $\leftarrow \mathcal{J}_{NA}$(Winning-Agent($j$)) $\cup \{j\}$
  end if
  for $r' \in$ Dependent-Relays(j) **do**
    Dependent-Tasks(r') $\leftarrow$ Dependent-Tasks(r') $\setminus \{j\}$
    if Dependent-Tasks(r') $= \emptyset$ **then**
      $\mathcal{R} \leftarrow \mathcal{R} \setminus \{r'\}$
      $\mathcal{A} \leftarrow \mathcal{A} \setminus \{r'\}$ **if** $r' \in \mathcal{A}$
    end if
  end for
  Dependent-Relays(j) $\leftarrow \emptyset$
end for
return $(\mathcal{J}, \mathcal{R}, \mathcal{A}, \mathcal{J}_{NA})$

Algorithm 3 CREATE-RELAYS($\mathcal{I}$, $\mathcal{J}$, $\mathcal{R}$, $\mathcal{A}$, $\mathcal{J}_{NA}$)

$\mathcal{R}' \leftarrow \mathcal{R}$
for ($j \in \mathcal{A}$) & (Dependent-Relays(j) $= \emptyset$) **do**
  $(\mathcal{N}_B, \mathcal{N}_S) \leftarrow$ PREDICT-NETWORK(t_j, $\mathcal{I}$, $\mathcal{J}$, $\mathcal{R}$, $\mathcal{A}$)
  for $\mathcal{N}_{s_k} \in \mathcal{N}_S$ **do**
    $(i_B, i_{s_k}) \leftarrow$ Closest-Agents($\mathcal{N}_B$, $\mathcal{N}_{s_k}$)
    $n_R = \lfloor ||i_B - i_{s_k}||_2 / R_{COMM} \rfloor$
    if $n_R < |\mathcal{I}|$ **then**
      $\mathcal{R}_{NEW} \leftarrow$ Make-Relays(i_B, i_{s_k}, n_R)
      $\mathcal{J}' \leftarrow$ Dependent-Tasks($\mathcal{N}_{s_k}$)
      Start-Time($\mathcal{R}_{NEW}$) $\leftarrow \min_{j' \in \mathcal{J}'}$(Start-Time($j'$))
      End-Time($\mathcal{R}_{NEW}$) $\leftarrow \max_{j' \in \mathcal{J}'}$(End-Time($j'$))
      Score($\mathcal{R}_{NEW}$) $\leftarrow \frac{1}{n_R} \sum_{j' \in \mathcal{J}'}$ Score(j')
      $(\mathcal{R}_{NEW}, \mathcal{J}) \leftarrow$ Set-Dependencies($\mathcal{R}_{NEW}$, $\mathcal{J}'$)
      Broadcast-Relays($\mathcal{R}_{NEW}$)
      $\mathcal{R}' \leftarrow \mathcal{R}' \cup \mathcal{R}_{NEW}$
    else
      $\mathcal{A} \leftarrow \mathcal{A} \setminus \{j\}$
      $\mathcal{J}_{NA}$(Winning-Agent(j)) $\leftarrow \mathcal{J}_{NA}$(Winning-Agent($j$)) $\cup \{j\}$
    end if
  end for
end for
return $(\mathcal{J}, \mathcal{R}', \mathcal{A}, \mathcal{J}_{NA})$

Algorithm 4 PREDICT-NETWORK(t, $\mathcal{I}$, $\mathcal{J}$, $\mathcal{R}$, $\mathcal{A}$)

1: $(\mathcal{I}', \mathcal{J}') \leftarrow$ Active-Agents(t, $\mathcal{I}$, $\mathcal{J}$, $\mathcal{R}$, $\mathcal{A}$)
2: Positions($\mathcal{I}'$) $\leftarrow$ Predict-Positions(t, $\mathcal{I}'$, $\mathcal{J}$, $\mathcal{R}$, $\mathcal{A}$)
3: $(\mathcal{N}_B, \mathcal{N}_S) \leftarrow$ Check-Connectivity(Positions($\mathcal{I}'$))
4: $(\mathcal{N}_B, \mathcal{N}_S) \leftarrow$ Add-Task-Info($\mathcal{J}'$)
5: **return** $(\mathcal{N}_B, \mathcal{N}_S)$[a]

task interval for the tasks they are connecting. This information can all be obtained from the task descriptions, the winning agent lists, the winning bid lists and the winning time lists. Note that in the mission execution, agents performing relay tasks do not actually receive a score, but they do incur fuel penalties. In a sense, the relay scores are "virtual scores", but by setting these scores to be the sum of the connected tasks divided by the number of grouped relays, the total mission scores obtained (scores for connected tasks minus all fuel usage) will never be negative.

- During the DELETE-RELAY phase, any unassigned relays are deleted along with all their grouped relays and dependencies. The task to which this relay belonged must then be released from the winning agent's bundle and that agent is not allowed to bid on that particular task again. This step guarantees algorithm convergence, since eventually any infeasible task will be on all agents' forbidden lists, or the agents will not be able to accommodate any new tasks. However, one subtlety of the algorithm is that the agent is allowed to hold on to the task for the subsequent iteration with some probability p that is proportional to the task value for that agent (normalized against the maximum obtainable task score). This stochastic aspect of the algorithm breaks the symmetry associated with the iterations and prevents all agents from releasing their tasks simultaneously, significantly improving algorithm convergence.

- Another algorithm subtlety is that the network prediction step (Algorithm 4) only involves agents that are currently executing tasks at time t (termed "active agents"). This is because "inactive" agents are subject to change their schedules to satisfy relay tasks, and this would invalidate the network prediction if they were included. The network prediction is only performed for the task execution time as opposed to most common approaches that involve discretizing time over the duration of the mission. By performing network detection only at select crucial times the computation associated with this algorithm remains tractable.

- The CBBA, DELETE-RELAY, and CREATE-RELAYS functions can all be executed by each agent in a distributed fashion, preserving the scalability of the original distributed algorithm. The information that needs to be broadcast and received by agents includes the location for new relays, deleted relay information, and the updated assignment information throughout the process.

The CBBA with Relays algorithm is guaranteed to converge, runs in real-time, and ensures a strongly connected network while tasks are being executed. The next section describes a conservative version of this algorithm that only involves predicting the network but not creating relays. The results show that CBBA with Relays outperforms this CBBA with Network Prediction only, as well as the Baseline CBBA.

[a] $\mathcal{N}_B$ is the set of active agents connected to the base station at time t; $\mathcal{N}_S$ are sets of subnetworks consisting of active agents connected with each other, but who are not connected to the base station.

III.B. CBBA with Network Prediction: Algorithm Description

The CBBA with Network Prediction algorithm is presented in Algorithm 5, and involves conservatively dropping tasks that are likely to cause network disconnects. The algorithm obtains an initial assignment using CBBA. For all assigned tasks, the network is predicted at the task execution time, and sets for agents connected to the base station ($\mathcal{N}_B$) and for those within their own subnetworks ($\mathcal{N}_S$) are returned. If the task will cause a network disconnect (i.e. the winning agent does not belong in $\mathcal{N}_B$ at the task execution time), the agent must drop that task and is not allowed to bid on it again. The algorithm then iterates until the assignment remains constant and no more tasks are being dropped. The network prediction method used in the CBBA with Network Prediction algorithm is more optimistic than the CBBA with Relays network prediction, since agents will not be changing their schedules to accommodate relay tasks and thus can all be included in the prediction. This algorithm is also guaranteed to converge, since infeasible tasks will eventually be on all agents' forbidden lists or all agents will not be able to add further tasks to their assignment vectors. This algorithm can also be executed in a distributed fashion, maintaining the scalability properties of the Baseline CBBA algorithm. A block diagram illustrating the CBBA with Network Prediction algorithm is provided in Figure 5(b).

Algorithm 5 CBBA-Networks($\mathcal{I}$, $\mathcal{J}$)

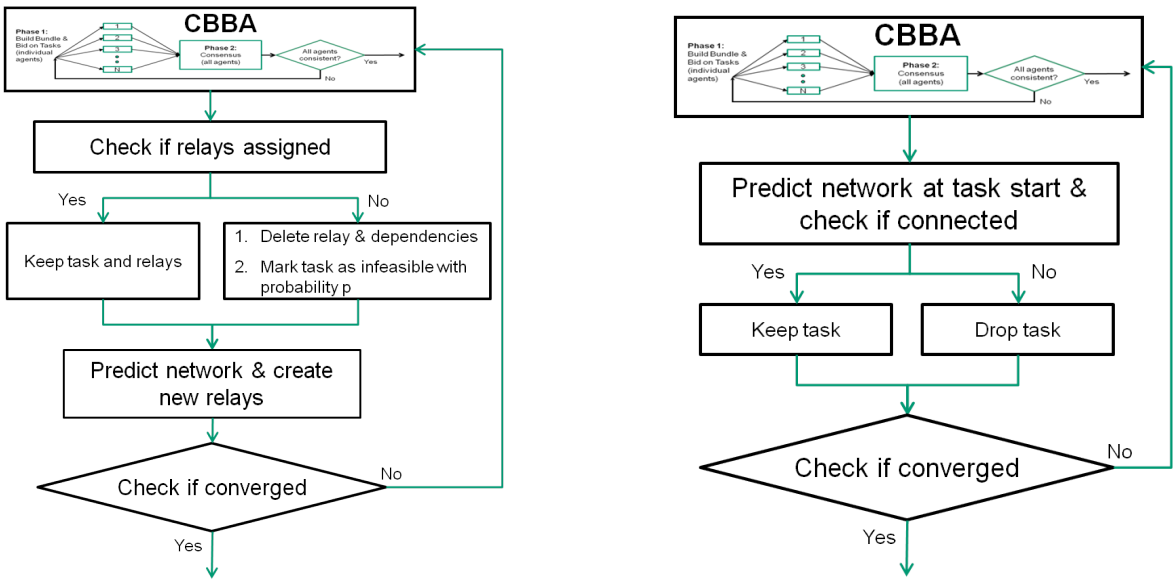
$\mathcal{A} \leftarrow \emptyset$; $\mathcal{J}_{NA}(i) \leftarrow \emptyset, \forall i \in \mathcal{I}$
converged $\leftarrow$ false
while $\neg$ *converged* **do**
  $\mathcal{A}' \leftarrow$ CBBA($\mathcal{I}$, $\mathcal{J}$, $\mathcal{A}$, $\mathcal{J}_{NA}$) (See [15,35] for details)
  ($\mathcal{A}'$, $\mathcal{J}_{NA}$) $\leftarrow$ Drop-Disconnecting-Tasks($\mathcal{I}$, $\mathcal{J}$, $\mathcal{A}'$, $\mathcal{J}_{NA}$)
  if $\mathcal{A}' = \mathcal{A}$ **then**
    converged $\leftarrow$ true
  else
    $\mathcal{A} \leftarrow \mathcal{A}'$
  end if
end while
return $\mathcal{A}$

Algorithm 6 Drop-Disconnecting-Tasks($\mathcal{I}$, $\mathcal{J}$, $\mathcal{A}$, $\mathcal{J}_{NA}$)

for $j \in \mathcal{A}$ **do**
  ($\mathcal{N}_B$, $\mathcal{N}_S$) $\leftarrow$ Predict-Network-Optimistic(t_j, $\mathcal{I}$, $\mathcal{J}$, $\mathcal{A}$)
  if Winning-Agent(j) $\notin \mathcal{N}_B$ **then**
    $\mathcal{A} \leftarrow \mathcal{A} \setminus \{j\}$
    $\mathcal{J}_{NA}$(Winning-Agent(j)) $\leftarrow \mathcal{J}_{NA}$(Winning-Agent($j$)) $\cup \{j\}$
  end if
end for
return ($\mathcal{A}$, $\mathcal{J}_{NA}$)

Algorithm 7 Predict-Network-Optimistic(t, $\mathcal{I}$, $\mathcal{J}$, $\mathcal{A}$)

Positions($\mathcal{I}$) $\leftarrow$ Predict-Positions(t, $\mathcal{I}$, $\mathcal{J}$, $\mathcal{R}$, $\mathcal{A}$)
($\mathcal{N}_B$, $\mathcal{N}_S$) $\leftarrow$ Check-Connectivity(Positions($\mathcal{I}$))
return ($\mathcal{N}_B$, $\mathcal{N}_S$)



(a) CBBA with Relays Algorithm

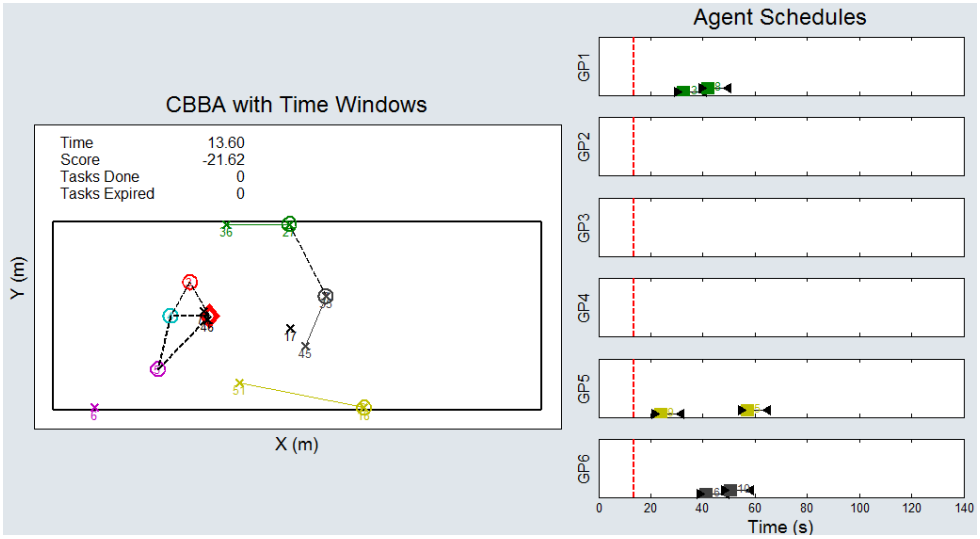
(b) CBBA with Network Prediction Algorithm

Figure 5. Block diagram overviews of the CBBA with Relays algorithm and the CBBA with Network Prediction algorithm

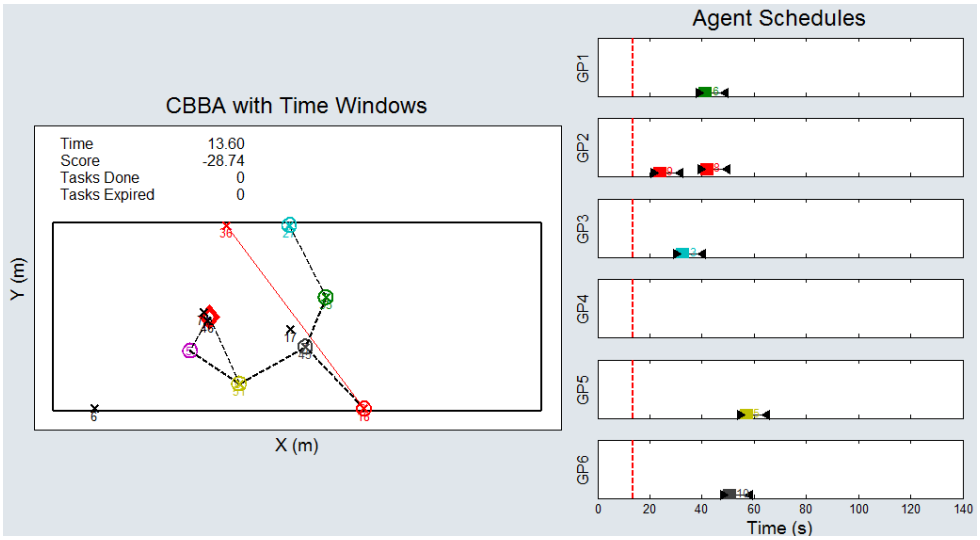
IV. Results and Discussion

To compare the performance of CBBA with Relays, CBBA with Network Prediction and the Baseline CBBA, a real-time simulation framework was created in MATLAB. The simulation architecture, depicted in Figure 4, consisted of the task allocation planner, a mission control center, a network detection algorithm, and vehicle managers. The mission control center supplied an initial set of tasks with time windows of validity to the agents, in addition to creating new pop-up tasks every 20 seconds. The task allocation planner received the dynamic list of tasks and allocated them to the agents, producing task lists for each agent. The network detection algorithm used the position of the vehicles to determine the network graph and the set of subnetworks at any given time. Finally, the vehicle manager consisted of a finite state machine which executed the tasks in each agent's task list. Screen shots of the simulation architecture are depicted in Figure 6.

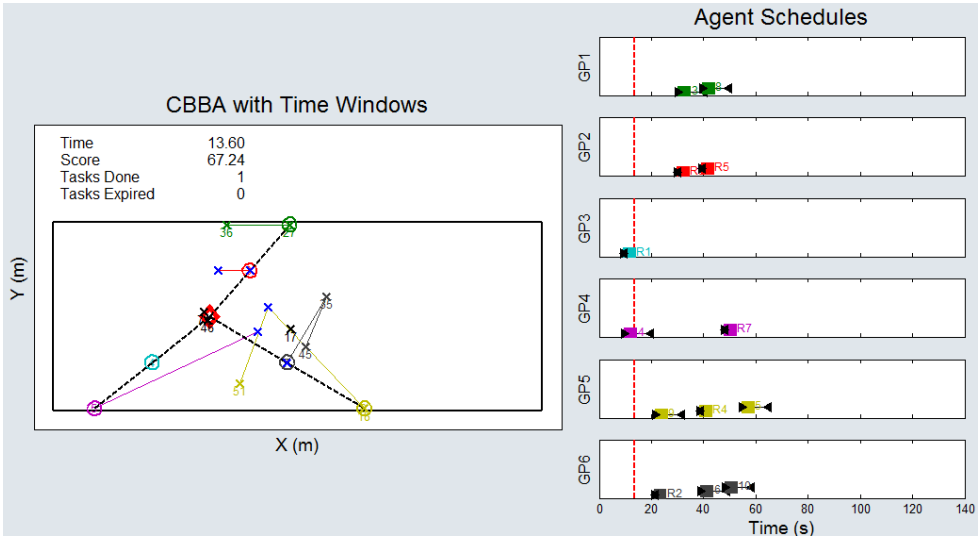
The scenario used to compare the algorithms included 6 agents, task durations of 5 seconds with 10 second time-windows for execution, 10 initial tasks, and 2 new pop-up tasks every 20 seconds. The task distribution is shown in Figure 8(a) and consisted of a Gaussian distribution with a mean and standard deviation of 3.5 m, centered at the base station. The value of the tasks was set to increase linearly with the distance from the base station (Figure 8(b)) up to a certain limit, which represented that information further from the base is more valuable since it is less likely to be known. The vehicle's speeds were all set to 0.5 m/s and the replan rate was every 20 seconds. In this scenario, agents were only able to execute tasks and receive rewards for them if they were connected to the base station at the time of task execution. If they were not connected to the base, then they had to abandon the corresponding task and move to the next one in their task list. Figure 6 presents screen shots of Baseline CBBA, CBBA with Network Prediction, and CBBA with Relays, for a scenario with a normalized communication radius of 20% of the theater of operation. The plots show the agent paths (left) and schedules (right) as well as the current network connectivity (black dotted lines). The relay tasks are depicted with blue x's, the regular tasks with black x's, and the base station is shown as a red diamond. Figure 6(a) shows that the Baseline CBBA causes significant network disconnects. Figure 6(b) shows that CBBA with



(a) Baseline CBBA



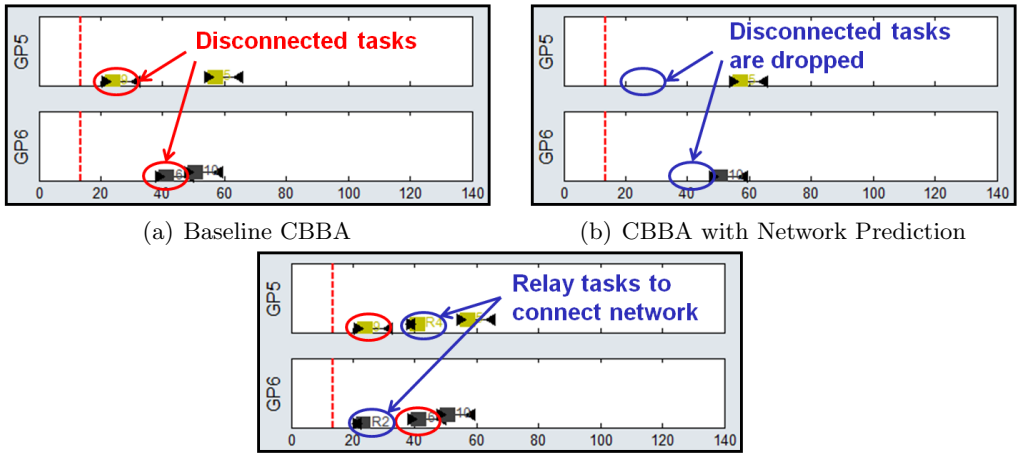
(b) CBBA with Network Prediction



(c) CBBA with Relays

Figure 6. Simulation screen shots of the different algorithms

Network Prediction improves the mission performance by reducing network disconnects but is still very conservative in the tasks it schedules. Figure 6(c) shows the CBBA with Relays algorithm. Note that in this scenario, agents bid on relay tasks if they can, filling up their empty schedules and allowing other agents to perform farther tasks. In particular, observe the schedules of agents 5 and 6 in Figures 6(a), 6(b), and 6(c) (the relevant portions of the figures are displayed in Figure 7). In the baseline case, agents 5 and 6 bid on 2 tasks each, however, 2 of these tasks will cause network disconnects (Figure 7(a)). In the CBBA with Network Prediction case, the 2 agents identify the tasks that will cause network disconnects and drop them from their schedules (Figure 7(b)). In the CBBA with Relays case, agents 5 and 6 propose relays for their disconnecting tasks, and realize that they are able to accommodate these relays for each other in addition to their other tasks. Therefore they are able to keep the previously disconnecting tasks achieving a higher overall mission score.

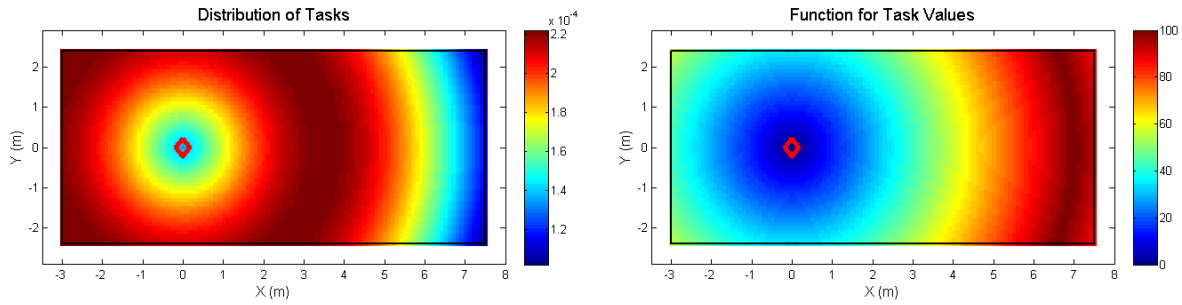


(a) Baseline CBBA

(b) CBBA with Network Prediction

(c) CBBA with Relays

Figure 7. Blow-ups of schedules for agents 5 and 6 from the simulation screen shots in Figure 6

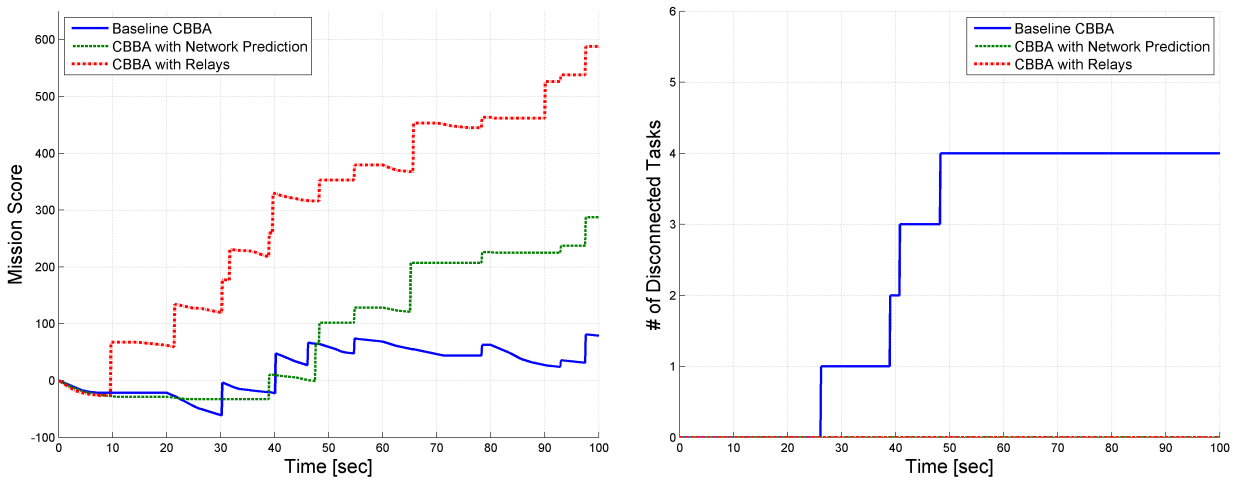


(a) Probability Distribution for Tasks

(b) Function for Task Values

Figure 8. Task specifications over the theater of operation for the simulation scenario. The location of the base station is depicted by the red diamond.

Figure 9 shows plots of the mission score as a function of time as well as the number of disconnected tasks during execution as a function of time for the scenario described above. As seen in the plots, CBBA with Relays clearly outperforms both the CBBA with Network Prediction algorithm and the Baseline CBBA, achieving a higher score at all times. Both extensions to CBBA ensure network connectivity during task execution achieving zero disconnected tasks throughout the mission.



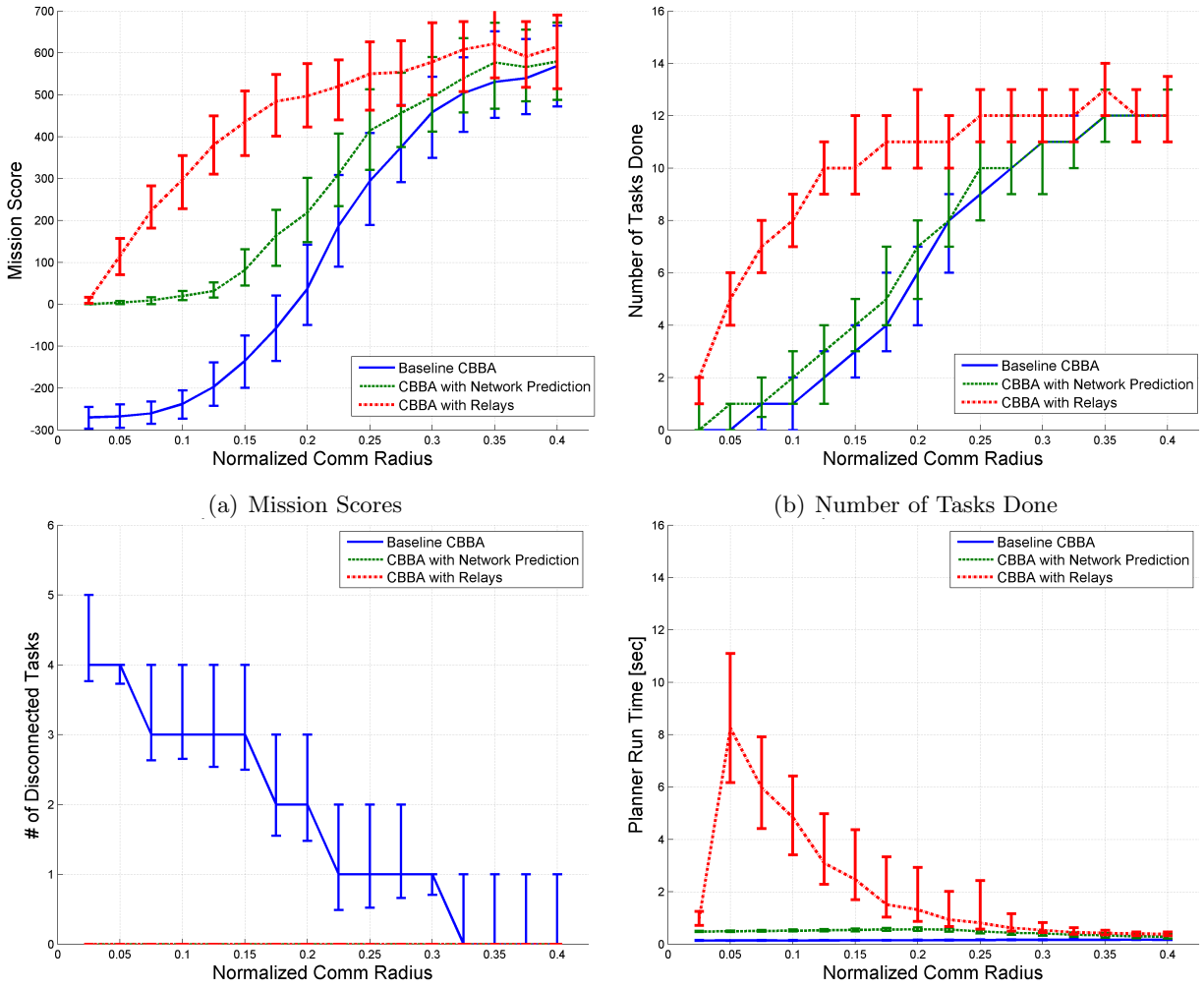
(a) Mission Scores

(b) Number of Disconnected Tasks

Figure 9. Single simulation run comparing Baseline CBBA, CBBA with Network Prediction, and CBBA with Relays, for a 20% normalized communication radius

To further analyze the performance of CBBA with relays, a Monte Carlo simulation was implemented, utilizing the same scenario described above but varying the normalized communication radius. Each test scenario was executed 100 times and the resulting medians with 25% and 75% error bars are depicted in Figure 10 for the different mission statistics. The mission score, the number of tasks done, the number of disconnected tasks and the planner run-time were compared for several different values of the communication radius. Figure 10 shows that CBBA with Relays achieves higher mission performance than the other two approaches, with higher scores and greater number of tasks done. Both CBBA with Network Prediction and the baseline CBBA achieve similar number of tasks performed, however, in the baseline CBBA agents attempt to execute tasks which will cause disconnects, and thus waste fuel without being able to perform the far tasks leading to lower mission scores. Both CBBA with Relays and CBBA with Network Prediction ensure that no agents are disconnected from the base during task execution, as seen in the lower left figure. The planner run-time for CBBA with Relays is higher than that for the other two algorithms, and is highest when several relays must be assigned. It drops off as the connectivity improves and at really low communication radii, since the number of relays required is greater than the available agents, making the tasks infeasible immediately, thus lowering the plan time. At a normalized communication radii higher than 0.3 (i.e. 30% of the theater of operation), the network remains mostly connected and all the algorithms achieve similar performance. These results demonstrate that CBBA with Relays successfully addresses the problem of network disconnects, achieving high performance in weak communication environments.

To validate the real-time applicability and performance of the proposed algorithms, hardware experiments were conducted at MIT's *Real-time indoor Autonomous Vehicle test ENvironment* (RAVEN),[36,37] shown in Figure 11(a). This indoor flight facility is equipped with a motion-capture system which yields accurate, high-bandwidth position and attitude data for all tracked vehicles within the flight volume. For these missions, the robotic hardware platform consisted of iRobot's Create (Figure 11(b)), equipped with an xBee-PRO wireless module for communication. Six of these hardware ground agents were used in each mission, and the 3 algorithms were compared for the scenario described above with a communication radius of 15% of the flight environment. Figure 11(c) depicts a mission snapshot for the CBBA with Relays case, showing the coordinated behavior for the 6-agent team. Here agents 1 and 6 act as relays for agent 2, and agent 5 satisfies the relay tasks for agents 3 and 4. Using the CBBA with Relays algorithm, the agents are able



(a) Mission Scores

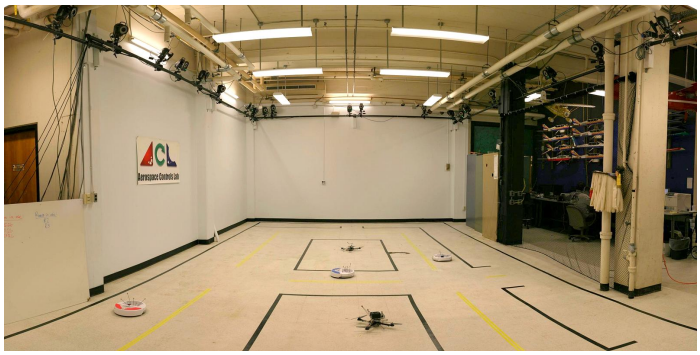
(b) Number of Tasks Done

(c) Average Number of Disconnected Tasks

(d) Average Planner Run Time

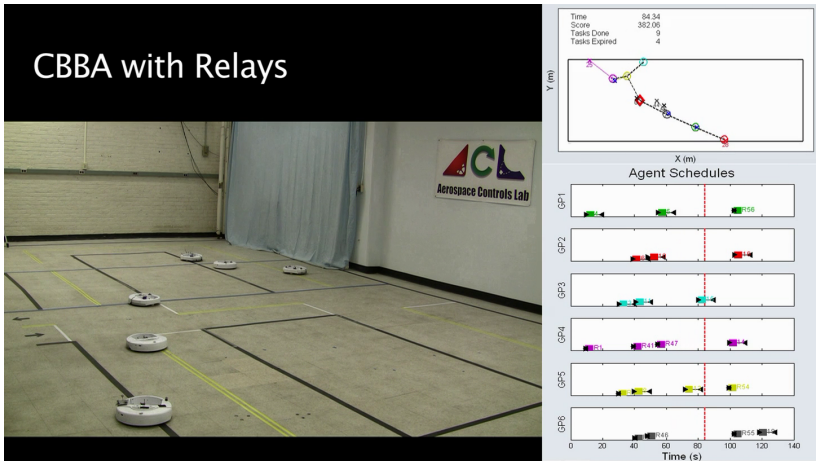
Figure 10. Monte Carlo simulation results comparing the performance of Baseline CBBA, CBBA with Network Prediction, and CBBA with Relays, as a function of normalized communication radius

to reach the farther high-value tasks, thus improving mission performance. The hardware results for this mission scenario are shown in Figure 12 and resemble those previously presented in Figure 9. The mission scores are highest using the CBBA with Relays algorithm, and both extensions of CBBA ensure that no tasks are disconnected throughout the mission. These hardware results demonstrate the real-time applicability of these algorithms and show that they are able to address the connectivity challenges for networked teams performing in limited connectivity environments.



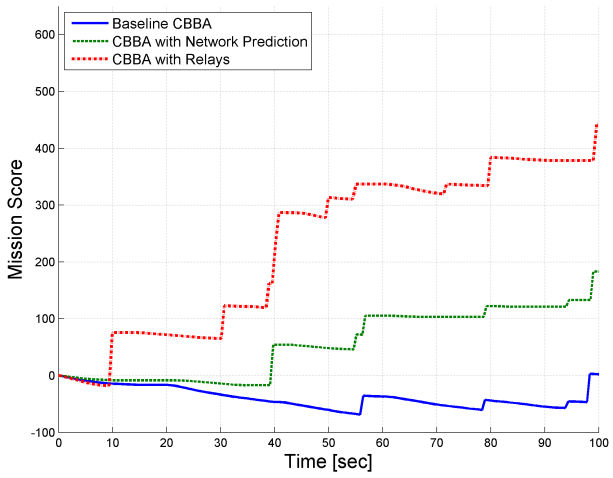
(a) MIT RAVEN (Aerospace Controls Lab)

(b) iRobot Create Hardware Platform

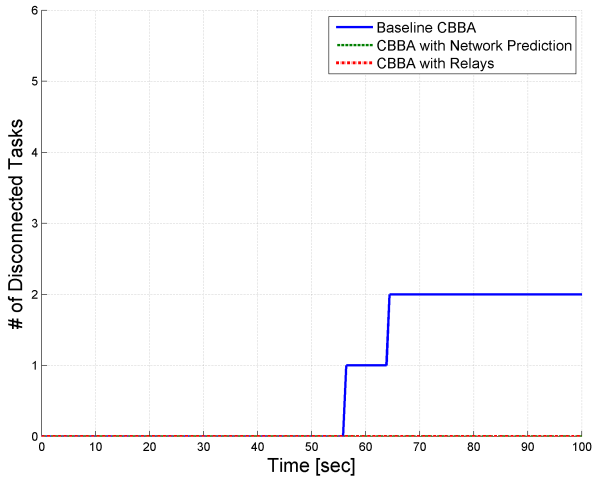


(c) 6-Agent Experimental Mission using CBBA with Relays

Figure 11. Real-time experimental mission setup to compare Baseline CBBA, CBBA with Network Prediction, and CBBA with Relays



(a) Mission Scores



(b) Number of Disconnected Tasks

Figure 12. Real-time experimental mission results comparing Baseline CBBA, CBBA with Network Prediction, and CBBA with Relays, for a 15% normalized communication radius

V. Conclusions

In this paper, we present two extensions to the Consensus-Based Bundle Algorithm (CBBA) designed to prevent and repair network disconnects for a team of heterogeneous agents performing in a dynamic environment. In CBBA with Network Prediction, agents improve mission performance over the Baseline CBBA by predicting potential network disconnects and conservatively dropping assignments. In the CBBA with Relays algorithm, varying network topologies and dynamic communication constraints are predicted, and relay tasks are created to strengthen the connectivity of the network. Key features of both algorithms are that they only predict the network topology at select mission critical times and that they leverage local information already available to each agent, making their execution distributed and polynomial-time, enabling real-time applications. Both algorithms ensure network connectivity during task execution, however, CBBA with Relays achieves higher mission performance with nominal increases in planner run time. In particular, by employing underutilized resources, CBBA with Relays improves network connectivity without limiting the scope of the active agents thus improving mission performance and reducing communication disconnects.

Acknowledgments

This work was sponsored (in part) by the AFOSR and USAF under grant (FA9550-08-1-0086) and MURI (FA9550-08-1-0356). The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the Air Force Office of Scientific Research or the U.S. Government. The third author gratefully acknowledges for financial support by Korea Agency for Defense Development Grant ADDR-401-101761.

References

[1] "Unmanned Aircraft Systems Roadmap, 2005-2030," Tech. rep., Office of the Secretary of Defense, August 2005.

[2] Bertsimas, D. and Weismantel, R., *Optimization over integers*, Dynamic Ideas Belmont, MA, 2005.

[3] Alighanbari, M., Kuwata, Y., and How, J., "Coordination and control of multiple UAVs with timing constraints and loitering," *American Control Conference*, Vol. 6, 4–6 June 2003, pp. 5311–5316.

[4] Dondo, R. and Cerdá, J., "An MILP framework for dynamic vehicle routing problems with time windows," *Latin American Applied Research*, Vol. 36, No. 4, 2006, pp. 255–261.

[5] Choi, H.-L., Whitten, A. K., and How, J. P., "Decentralized Task Allocation for Heterogeneous Teams with Cooperation Constraints," *American Control Conference (ACC)*, July 2010, pp. 3057–3062.

[6] Ponda, S., Redding, J., Choi, H.-L., How, J. P., Vavrina, M. A., and Vian, J., "Decentralized Planning for Complex Missions with Dynamic Communication Constraints," *American Control Conference (ACC)*, July 2010.

[7] Olfati-Saber, R. and Murray, R. M., "Consensus problems in networks of agents with switching topology and time-delays," *IEEE Trans. Automat. Contr.*, Vol. 49, no. 9, Sept. 2004, pp. 1520–1533.

[8] Olfati-Saber, R., Fax, J. A., and Murray, R. M., "Consensus and Cooperation in Networked Multi-Agent Systems," *IEEE Proceedings*, Vol. 95, No. 1, January 2007, pp. 215–233.

[9] Nguyen, H., Pezeshkian, N., Raymond, M., Gupta, A., and Spector, J., "Autonomous communication relays for tactical robots," *Proceedings of the International Conference on Advanced Robotics (ICAR)*, 2003.

[10] Ibrahim, A., Seddik, K., and Liu, K., "Improving connectivity via relays deployment in wireless sensor networks," 2007, pp. 1159–1163.

[11] Palat, R., Annamalau, A., and Reed, J., "Cooperative relaying for ad-hoc ground networks using swarm UAVs," *Military Communications Conference, 2005. MILCOM 2005. IEEE*, IEEE, 2005, pp. 1588–1594.

[12] Burdakov, O., Doherty, P., Holmberg, K., Kvarnstr
"om, J., and Olsson, P., "Positioning unmanned aerial vehicles as communication relays for surveillance tasks," *Proceedings of Robotics: Science and Systems, Seattle, USA*, 2009.

[13] Pinkney, M., Hampel, D., and DiPierro, S., "Unmanned aerial vehicle (UAV) communications relay," *Military Communications Conference, 1996. MILCOM'96, Conference Proceedings, IEEE*, Vol. 1, IEEE, pp. 47–51.

[14] Dixon, C. and Frew, E., "Maintaining optimal communication chains in robotic sensor networks using mobility control," *Mobile Networks and Applications*, Vol. 14, No. 3, 2009, pp. 281–291.
[15] Choi, H.-L., Brunet, L., and How, J. P., "Consensus-Based Decentralized Auctions for Robust Task Allocation," *IEEE Trans. on Robotics*, Vol. 25 (4), 2009, pp. 912 – 926.
[16] McLain, T. W. and Beard, R. W., "Coordination Variables, Coordination Functions, and Cooperative-Timing Missions," *Journal of Guidance, Control, and Dynamics*, Vol. 28(1), 2005, pp. 150–161.
[17] Castanon, D. A. and Wu, C., "Distributed algorithms for dynamic reassignment," *IEEE Conference on Decision and Control*, Vol. 1, 9-12 Dec. 2003, pp. 13–18 Vol.1.
[18] Curtis, J. and Murphey, R., "Simultaneous Area Search and Task Assignment for a Team of Cooperative Agents," *Proceedings of AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2003.
[19] Shima, T., Rasmussen, S. J., and Chandler, P., "UAV team decision and control using efficient collaborative estimation," *Proceedings of the American Control Conference*, 8-10 June 2005, pp. 4107–4112 vol. 6.
[20] Ren, W., Beard, R. W., and Kingston, D. B., "Multi-agent Kalman consensus with relative uncertainty," *Proceedings of the American Control Conference*, 8-10 June 2005, pp. 1865–1870 vol. 3.
[21] Ren, W. and Beard, R., "Consensus seeking in multiagent systems under dynamically changing interaction topologies," *IEEE Transactions on Automatic Control*, Vol. 50, No. 5, May 2005, pp. 655–661.
[22] Olfati-Saber, R. and Murray, R. M., "Consensus Problems in Networks of Agents with Switching Topology and Time-Delays," *IEEE Transactions on Automatic Control*, Vol. 49(9), 2004, pp. 1520–1533.
[23] Alighanbari, M. and How, J. P., "Unbiased Kalman Consensus Algorithm," *Journal of Aerospace Computing Information and Control*, Vol. 5, No. 9, 2008, pp. 209–311.
[24] Moallemi, C. C. and Roy, B. V., "Consensus Propagation," *IEEE Transactions on Information Theory*, Vol. 52(11), 2006, pp. 4753–4766.
[25] Olshevsky, A. and Tsitsiklis, J. N., "Convergence Speed in Distributed Consensus and Averaging," *Proceedings of the 45th IEEE Conference on Decision and Control*, 2006.
[26] Ren, W., Beard, R. W., and Atkins, E. M., "Information consensus in multivehicle control," *IEEE Control Systems Magazine*, Vol. 27(2), 2007, pp. 71–82.
[27] Hatano, Y. and Mesbahi, M., "Agreement over Random Networks," *43rd IEEE Conference on Decision and Control*, 2004.
[28] Wu, C. W., "Synchronization and Convergence of Linear Dynamics in Random Directed Networks," *IEEE Transactions on Automatic Control*, Vol. 51, No. 7, 2006, pp. 1207?–1210.
[29] Tahbaz-Salehi, A. and Jadbabaie, A., "On Consensus Over Random Networks," *44th Annual Allerton Conference*, 2006.
[30] Alighanbari, M. and How, J. P., "Decentralized Task Assignment for Unmanned Aerial Vehicles," *Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference*, 2005.
[31] Sariel, S. and Balch, T., "Real Time Auction Based Allocation of Tasks for Multi-Robot Exploration Problem in Dynamic Environments," *Proceedings of the AIAA Workshop on Integrating Planning Into Scheduling*, 2005.
[32] Ahmed, A., Patel, A., Brown, T., Ham, M., Jang, M., and Agha, G., "Task assignment for a physical agent team via a dynamic forward/reverse auction mechanism," *International Conference on Integration of Knowledge Intensive Multi-Agent Systems*, 2005.
[33] Bertuccelli, L., Choi, H., Cho, P., and How, J., "Real-time Multi-UAV Task Assignment in Dynamic and Uncertain Environments," .
[34] Ponda, S., Choi, H.-L., and How, J. P., "Predictive planning for heterogeneous human-robot teams," *AIAA Infotech@Aerospace*, April 2010.
[35] Johnson, L. B., Ponda, S., Choi, H.-L., and How, J. P., "Improving the Efficiency of a Decentralized Tasking Algorithm for UAV Teams with Asynchronous Communications," *AIAA Guidance, Navigation, and Control Conference (GNC)*, August 2010.
[36] Valenti, M., Bethke, B., Fiore, G., How, J., and Feron, E., "Indoor Multi-Vehicle Flight Testbed for Fault Detection, Isolation, and Recovery," *Proceedings of the AIAA Guidance, Navigation, and Control Conference and Exhibit*, Keystone, CO, August 2006.
[37] J. How and B. Bethke and A. Frank and D. Dale and J. Vian, "Real-Time Indoor Autonomous Vehicle Test Environment," *IEEE Control Systems Magazine*, Vol. 28, No. 2, April 2008, pp. 51–64.