
DCOPs Meet the Real World: Exploring Unknown Reward Matrices with Applications to Mobile Sensor Networks

Manish Jain, Matthew Taylor, Milind Tambe
Teamcore Research Group
University of Southern California
Los Angeles, CA 90089
{manish.jain, taylorm, tambe}@usc.edu

Makoto Yokoo
Kyushu University
Fukuoka 812-8581, Japan
yokoo@is.kyushu-u.ac.jp

Abstract

Buoyed by the recent successes in the area of *distributed constraint optimization problems* (DCOPs), this paper addresses challenges faced when applying DCOPs to real-world domains. This expedition reveals that three fundamental challenges must be addressed for a large class of real-world domains, requiring design of novel DCOP algorithms. First, in many domains, agents do not know the initial payoff matrix and must explore the environment to determine rewards associated with different variable settings. Second, the agents have a goal to maximize the total accumulated reward rather than the instantaneous reward at the end of the run. Third, limited task-time horizons disallow agents the luxury of full exploration of their environment and payoff matrices. We propose and implement a set of novel algorithms and provide positive experimental results. At their core, these new algorithms interweave decision-theoretic approaches to explore the environment within the limited time horizon with the DCOP-mandated coordination. In addition to simulation results, we implement these algorithms on robots, deploying DCOPs on a distributed mobile sensor network – illustrating the benefits of DCOPs in the real world.

1 Introduction

Distributed constraint optimization problems (DCOPs) [10, 12, 20] are a class of problems where cooperative agents must coordinate to maximize some reward. Examples include multiagent plan coordination [4], sensor networks [8, 20], meeting scheduling [16] and RoboCup soccer [18]. A team of agents coordinate their individual actions within a DCOP to achieve joint goals, but the utility of an agent’s action depends on the action choices of a subset of the other agents; DCOPs are thus ideally suited when agents must coordinate via local interactions. Significant progress has been achieved in design and analysis of globally optimal DCOPs algorithms (c.f., Adopt [12], DPOP [16], and OptAPO [10]). However, given that DCOPs are NP-Hard [12], significant communication and computation overheads result in attempting to solve them optimally, which motivates the need for locally optimal algorithms. Such locally optimal algorithms that have been shown to scale to much larger tasks in practice [20, 15, 19].

Given their recent progress, DCOPs are now ideally poised to tackle real-world applications. Motivated by this objective, we target a large class of real-world distributed sensor network applications, that include tasks such as AUVs (Autonomous Underwater Vehicles) [21] used for surveying underwater structures, UAVs (unmanned air vehicles) to measure environmental phenomena [2], and small mobile robots that establish a communication network. Our study reveals that three novel challenges must be addressed in applying DCOP algorithms to these domains. First, agents in these domains do not know the initial payoff matrix and must explore the environment to determine rewards associated with different variable settings. All payoffs are dependent on agents' joint actions, requiring them to coordinate in their exploration. Second, the agents must maximize the total accumulated reward rather than the instantaneous reward at the end of the run. Third, agents face a limited task-time horizon, requiring efficient exploration. These challenges disallow direct application of current DCOP algorithms which implicitly assume that all agents have knowledge of the full payoff matrix. Furthermore, agents cannot fully explore their environment to learn the full payoff matrices and then run a globally optimal algorithm. The time horizon is much too short for such a sequential phase of exploration followed by optimization; indeed, interleaving these phases may improve accumulated reward during exploration.

To address these challenges, this paper proposes novel DCOP algorithms based on two key ideas. First, as mentioned above, these DCOP algorithms must seamlessly interleave *distributed exploration* and *distributed exploitation* phases. Second, DCOP algorithms may need a range of exploration strategies in their arsenal, each potentially useful in a different setting. For example, in a *one-step* strategy, an agent deciding to explore examines one unknown payoff matrix value at a time and triggers optimization, enabling fine-grained interleaving of exploration and exploitation. In contrast, a *multi-step* exploration strategy allows a single agent to scout out multiple payoff matrix values and then select the best observed during the exploration phase. Here a decision-theoretic approach uses knowledge of reward distribution to compute the optimal number of unknown values to explore and the expected reward gain. However, this approach forces a coarser-grained integration where a single agent in a neighborhood of agents explores for multiple steps before triggering distributed optimization. We also introduce a *hybrid* approach which attempts to combine the strengths of the two strategies. In particular, agents compute their expected gain as in a multi-step exploration strategy, but trigger optimization at each time step, enabling fine-grained interleaving of exploration and exploitation. Using these key ideas, we provide a family of five novel DCOP algorithms.

Given this diverse family of algorithms, it is crucial to understand if particular algorithms dominate others in key circumstances. Our empirical tests are based on a novel, concrete, real-world domain in which agents must maximize an accumulated signal strength in a mobile sensor network within a set time limit. Our algorithms enable agents to reason about movement policies in order to improve their signal strength over time and increase the reliability of the network. To perform such optimization efficiently, we model this problem as a DCOP with the novel extensions discussed above.

Our new DCOP algorithms are implemented not only on simulated agents, but on physical robots as well. Apart from early work on distributed constraint reasoning by Lesser et al. [8], this is the first application of DCOPs on physical robots with a demonstrated improvement in performance in a real world problem. Furthermore, the simulations allow us to gather experimental results quickly while changing parameters of the task (such as the number of robots in the network and length of the experiments) to further evaluate our algorithms.

Key results from our experiments include: (i) algorithms based on the hybrid strategy dominate in most circumstances and (ii) one-step strategies are sufficient when networks are fully connected or even superior when few optimization movements are allowed. Furthermore, we compare to an algorithm which is given the DCOP reward matrix at the beginning of each experiment and find that our algorithms are able to achieve signal improvements of up to 80% of this "omniscient" algorithm.

These results combine to show that DCOP algorithms can be modified to work in environments where: on-line reward is critical, rewards are initially unknown, the algorithms are fully distributed, and using physical hardware. Combined, these results strongly suggest that DCOPs are a powerful and appropriate solution technique for complex multi-agent problems in the real world.

2 Background

This section of the paper first defines the mobile sensor network optimization task. Section 2.2 discusses the DCOP formulation and why it is useful for this problem domain.

2.1 Problem Domain

This paper focuses on tasks in which agents do not know their initial rewards, there is a fixed time horizon, and agents are evaluated during optimization (i.e., the on-line reward is critical). One such problem with these characteristics is a *wireless sensor network* (c.f., Akyildiz et al. [1]), where a common goal is to monitor a region of the world and report when interesting objects or events are perceived. Many multiagent problems share these characters; problems in this class include aerial surveillance [2] and underwater agent coordination [21].

Sensors in this paper are assumed to have movement abilities. Rather than framing the problem as an ad hoc mobile wireless network, we take the topology of the network as fixed under the assumption that humans have placed the robots in reasonable positions. We also assume that each sensor knows its *neighbors*, the sensors with which it can directly communicate. For example, during natural disasters, rescue personnel may quickly place such mobile sensors around a disaster site to relay information about endangered humans, fires, etc. It will then be critical for the sensors to quickly optimize the network’s signal strength to ensure reliable and effective communication.

Radio communication, commonly used in wireless sensor networks, has a predictable signal strength based on the inverse square of the distance between transmitter and receiver.¹ However, in urban or indoor settings, obstacles often interrupt line of sight communication, creating a *multi-path* setting. Scattering, reflection, and diffraction of radio waves make it very difficult to predict the optimal placement of sensors in a network. Constructive and destructive interference of the radio waves, known as the *small scale fading effect*, results in significant signal strength differences over small distances.

This paper concentrates on settings where the sensors are not line of sight. Due to small scale fading, the signal strength between two locations separated by a distance of at least $\frac{1}{2}$ of a wavelength is uncorrelated. Put another way, if a sensor moves $\frac{1}{2}$ of a wavelength, it will measure a new signal strength from each of its neighbors, where each signal can be modeled as an independent random number drawn from some distribution [7]. Our initial experiments suggest a Normal distribution, but our algorithms are distribution-independent and other distributions can easily be substituted.

We assume in this study that sensors have been placed in reasonable locations so that no sensor is disconnected from the network. Further, we assume that nodes do not fail, they are not malicious, and communication between neighbors is reliable. Given a network topology and length of the experiment (time T), our goal is to maximize the signal strength in the network over this time. Using l as our index over network links, we want to maximize:

$$\sum_{0 \leq t \leq T} \sum_{l \in \text{network}} \text{SignalStrength}(l, t)$$

by allowing agents to take small movements without changing the overall network topology. We discretize the experiment and the agents’ decision making into synchronized *rounds*. A round ends after all agents perform the required computation, finish communication, and move to a new location (if desired). The length of a round in our distributed setting is dominated by the robot movement time, which is much longer than either the computation of any algorithms in this paper or the communication time.²

Experiments in this paper use results from a set of Create robots from iRobot and from a custom simulator built to mimic properties of the Create. The Creates have 2 actuators which can be controlled.

¹For an in-depth discussion of signal strength propagation, we refer interested readers to more complete treatments elsewhere [13].

²If agents converge upon a final configuration before the test ends, no robots will move within a round. However, the length of the round does not vary, but remains the time necessary for a robot to move, calculate, and communicate. The length of a round determines how continuous time is discretized to measure signal strength in experiments but is not critical for measuring the relative performance of algorithms.



Figure 1: This photo shows a team of iRobot Creates, the platform used for physical tests of our algorithms. A more detailed view of the robot is shown in the inset.

They are capable of moving accurately in a straight line, while rotation introduces significant error. Each of these Creates have a wireless radio card provided by CenGen mounted on them. A fully charged battery is capable of running the Creates for approximately 3 hours. Figure 1 shows a team of the Create robots.

As the real world is continuous³ and Create robots have limited sensors, agents in our physical implementation are unable to determine their absolute location during experiments, but must instead rely on odometry to estimate relative location. Due to the high variance in signal strength over small distances and accumulated errors in odometry measurements, agents in some tasks may not be able to return to a previously observed signal strength. Thus, in this paper, we consider two distinct cases, as both may be valid, depending on the particular implementation of agent movement and available sensors. In the first case, each agent may either `stay` where it is, or `explore` by moving to a new location. In the second case, we assume that odometry errors can be ignored, which enables an agent to additionally execute the action `backtrack`, returning it to a previously explored location. In our physical implementation, the robots are able to `backtrack`, but we also run experiments assuming they cannot. Note that in this work we assume that agents are always able to explore a new state and *never* return to a previously visited state by selecting the `explore` action.

Mobile sensors must efficiently explore their surroundings to maximize the network signal strength as quickly as possible. If the goal was to maximize final signal strength and the agents had a small number of possible values, agents could explore the entire state space to populate the reward matrix and then use a traditional DCOP method to find an optimal setting of values (as discussed in the following section). However, because the goal is to maximize signal strength accumulated during the entire episode, fully exploring the state space would cause the agents to spend much of their time in highly sub-optimal configurations. Furthermore, in this work we assume that the number of states each robot can visit is large, making it impossible for a robot to explore all possible locations within the time of a single experiment, let alone the cross product of all possible locations.

2.2 DCOP Background

A DCOP consists of a set V of n variables, $\{x_1, x_2, \dots, x_n\}$, assigned to a set of agents, where each agent controls one variable's assignment. Variable x_i can take on any value from the discrete finite domain D_i . The goal is to choose values for the variables such that the sum over a set of binary constraints and associated payoff or reward functions, $f_{ij} : D_i \times D_j \rightarrow N$, is maximized. More specifically, find an assignment, A , s.t. $F(A)$ is maximized: $F(A) = \sum_{x_i, x_j \in V} f_{ij}(d_i, d_j)$, where $d_i \in D_i, d_j \in D_j$ and $x_i \leftarrow d_i, x_j \leftarrow d_j \in A$. Take the constraint graph in Figure 2 as an example. x_1, x_2 , and x_3 are variables, each with a domain of $\{0,1\}$ and the reward function as shown. If agents 2 and 3 choose the value 1, the agent pair gets a reward of 9. If agent 1 now chooses value 1 as well, the total solution quality of this complete assignment is 12, which is locally optimal

³As discussed in Section 4.1, we discretize the continuous world based on the wavelength of radio waves.

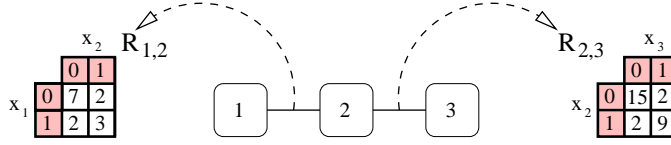


Figure 2: This figure depicts a three agent DCOP. Agents 1–3 each select a value for their corresponding variable ($x_1 - x_3$) from the domain $\{0, 1\}$. There are two sets of constraints which each define one of the two reward matrices.

as no single agent can change its value to improve its own reward (and that of the entire DCOP). $F((x_1 \leftarrow 0), (x_2 \leftarrow 0), (x_3 \leftarrow 0)) = 22$ and is globally optimal.

We use the mobile sensor network domain introduced in Section 2.1 as an experimental domain for our algorithms. The different robots in the network are the DCOP-aware agents. Communication links between robots represent constraints between agents and the signal strength obtained measures the reward of an assignment. The different physical positions of a robot constitute the domain of values possible for agents. An agent can accurately sense the signal strength between its current location and the current location of each of its neighbors only when it explores that particular location.

3 Solution Techniques

This section describes novel extensions to DCOP methods to tackle the class of problems outlined in the Introduction, using the mobile sensor network problem as a concrete example of one such problem. The distributed and multiagent nature of the problem makes DCOP an appropriate solution technique. The primary difference between this domain and traditional DCOP domains are: (1) firstly, the agents’ goal is to maximize the cumulative reward, (2) secondly, the agents do not know the reward matrix R , and (3) thirdly, the agents do not have time to explore all states. Thus, standard DCOP solvers cannot be directly applied to the type of problem discussed in this paper.

Given the inapplicability of globally optimal algorithms, we build on an existing locally optimal DCOP algorithms. The *Maximal Gain Messaging* (MGM) algorithm [15] and DSA [5] are natural candidates, but DSA has an additional probability parameter that must be set which has a significant impact on its performance [9]. While all the algorithms presented are in the framework of MGM, the key ideas of the paper can be embedded in any locally optimal DCOP framework. However, we keep the framework constant to ensure a fair comparison of the algorithms.

MGM-Omniscient: We first implement the *Maximal Gain Messaging* (MGM) algorithm [15] and artificially provide agents with all possible signal strengths. This represents an upper bound for us to compare with as the agents do not need to explore. Given such a matrix, any standard DCOP algorithm may quickly find a locally optimal solution for all agents. MGM-Omniscient defines a *round* as involving multiple broadcasts of *messages*. Every agent will broadcast its current value to all its neighbors at the beginning of the round. After the receipt of these messages, each agent broadcasts a *gain message* to all its neighbors that represents the maximum change in its local utility if it is allowed to act under the current context (i.e., values of neighboring agents). An agent is then allowed to act if its gain message is larger than all the gain messages it receives from all its neighbors (ties can be broken through variable ordering or another method) [19]. MGM-Omniscient belongs to the class of 1-optimal algorithms which have the property that no one agent can deviate from the proposed assignment and increase the net reward [15].

3.1 Static Estimation (SE) Algorithms

Our first set of solution techniques relies on selecting an action based on the probability of improving the signal strength on the next round. All are one-step approaches.

3.1.1 SE-Optimistic

In *Static Estimation with Optimistic Exploration* (SE-Optimistic), each agent assumes that if it moves to a new location, the signal strength between it and every neighbor will be maximized.

On every round, each agent bids its expected gain based on its current sum of signal strengths (R_c):

$$NumberLinks \times MaximumSignalStrength - R_c.$$

In each neighborhood, the agent with the highest bid is allowed to `explore` for the current round and other agents execute the `stay` action. Agents which have the lowest signal strengths will have the highest bid, similar to a 1-step greedy optimization algorithm. Because it will be rare to achieve maximal signal strengths to all neighbors, agents will typically continue to bid to explore on every round until the test concludes.

3.1.2 SE-Mean

Static Estimation with Mean-Aware Exploration (SE-Mean) modifies the previous algorithm to assume that visiting an unexplored state will result in the average signal strength to all neighbors (denoted μ) instead of the maximum. Agents therefore have an expected gain of:

$$NumberLinks \times \mu - R_c.$$

This modification to the previous algorithm causes agents to continue to greedily explore, but now agents will stop bidding to move once they achieve the average signal strength (averaged over all neighbors), allowing them to balance exploration with exploitation. Note that *MaximumReward* and μ can be defined initially as the reward distribution is known.

3.2 Balanced Exploration (BE) Algorithms

The objective of the *Balanced Exploration* (BE) approach, our second set of algorithms, is to allow each agent to explicitly estimate the value of exploration, which will depend on the following three things:

1. the number of timesteps left in the trial,
2. the distribution of the signal strengths (rather than just the maximum or the mean), and
3. the current signal strength of the agent, or the best explored signal strength if the agent can backtrack to a previously explored state.

The primary motivation for this class of algorithms is to allow the agents to more accurately estimate their gains. After agents calculate their expected gain from exploring or exploiting, each will decide whether to exploit its current signal strength or to bid to explore. As in MGM, the agent with the highest bid per neighborhood wins the ability to move.

3.2.1 BE-Backtrack

When an agent can execute the `backtrack` action, it will keep track of the location in which it has received the highest total signal strength (R_{best}). At any point, the agent may return to this location if the agent's neighbors have not moved. If one or more of the agent's neighbors have moved, return to this previous location will likely give a different total signal strength. Note that R_{best} will always be well defined, as an agent may always set $R_{best} = R_c$, its current total signal strength. The state of the agent can thus be defined as a 2-tuple of (R_{best}, T) consisting of the reward R_{best} , and the T time steps remaining in the current test.

The *Balanced Exploration with Backtracking* (BE-Backtrack) algorithm takes a multi-step approach, where an agent calculates $V(R_{best}, T)$, the expected utility of the agent if the current best backtrack location has a reward of R_{best} and T time steps remain in the current test.⁴ If the agent backtracked immediately to the location with reward R_{best} , again assuming that no neighbors move, the agent would accrue the utility R_{best} for the remainder of the experiment. Therefore, the value of backtracking will be

$$V_{back}(R_{best}, T) = R_{best}T.$$

⁴Note that throughout the discussion of the balanced exploration, there is a notion of *state*, which is different from an agent's *location*. For instance, the value of taking the `backtrack` action depends on the best reward seen (R_{best}) and the number of steps remaining in the episode, which describe the state. It does not depend on the current location, or the current signal strength. Thus $V_{back}(state) = V_{back}(R_{best}, T)$.

If the agent explores, it receives a reward based on the best location explored. Let the number of rounds for which the agent explores be t_e . An exploration policy would be in the form “explore for t_e rounds, `backtrack` to the best location found on round $t_e + 1$, and then `stay` in that location for the remainder of the experiment for t_s rounds, where $t_s = T - (t_e + 1)$.”

$V_{explore}(R_b, T)$ can be calculated by summing three separate components: the expected utility accrued while exploring for t_e steps, the utility accrued after exploration multiplied by the probability of finding a reward better than R_b , and finally the utility accrued after exploration multiplied by the probability of failing to find a reward better than R_b .

The first component will simply be $t_e \times \mu(n)$, where $\mu(n)$ is the average expected signal strength over n neighbors. The second component will depend on the probability of finding locations with a total signal strength higher than R_{best} , multiplied by the number of steps left in the trial. The expected best signal strength in this case will be described by the probability distribution:

$$\int_{x > R_{best}} x \times P(x, n, t_e) dx$$

where $P(x, n, t_e)$ gives the probability of x being the maximum sample among the t_e samples drawn when the agent has n neighbors and is defined as:

$$P(x, n, t_e) = t_e \times f(x, n) \times F(x, n)^{t_e - 1}$$

This n^{th} order statistics calculates the probability that x will be the maximum reward found in t_e values. n is the number of neighbors, $f(x, n)$ is the probability of drawing x as a sample, and $F(x, n)$ is the cumulative probability of drawing a sample less than or equal to x , defined as $\int_{y \leq x} f(y, n) dy$. Informally, $P(x, n, t_e)$ is calculated by drawing a sample x from any of the t_e samples with a probability $f(x, n)$, and drawing the rest of the $t_e - 1$ samples, such that their values are less than x , with a probability of $F(x, n)^{t_e - 1}$.

The third component will depend on how likely it is that we fail to discover a location better than R_{best} , multiplied by the number of steps left in the trial. After the agent explores, it will `backtrack` to the location that receives a total signal strength of R_{best} and the agent will receive this reward for the remaining t_s rounds. Again, the cumulative probability of drawing a sample less than or equal to R_{best} in t_e samples is defined as $F(R_{best})^{t_e}$, where $F(x)$ is defined as before.

Summing the three components, we find that $V_{explore}(R_{best}, T) =$

$$\max_{0 \leq t_e \leq T} \left\{ t_e \mu(n) + t_s \int_{x > R_b} x P(x, n, t_e) dx + t_s R_b F(R_b, n)^{t_e} \right\} \quad (1)$$

The value of t_e that maximizes $V_{explore}$ gives the number of exploration steps. The expected return of being in a location with T steps left, having previously seen a location with a total signal strength of R_{best} , is:

$$\max \left\{ V_{back}(R_{best}, T), V_{explore}(R_{best}, T) \right\}.$$

Having calculated the expected utility of `backtrack` and `explore`, the agent will select the action with the highest utility. Unless the action selected is `backtrack` and the current signal strength equals R_{best} , the agent will bid its expected utility, and the agent with the highest utility within each neighborhood will be allowed to move for t_e rounds, after which it `backtracks` to the location with the best found signal strength. BE-Backtrack dictates that the agent will not move after `backtracking` in the $(t_e + 1)^{th}$ round. However, if the agent’s neighbors later move, the agent may choose to explore rather than staying at its `backtracked` value.

Notice that when an agent’s neighbors explore and then `backtrack`, they could not have reduced the overall DCOP reward. In particular, the signal strength of an agent that has `backtracked` after exploring cannot be lower than its signal strength at the time it started exploring (although it may be lower during exploration). This is because only this agent was allowed to move in its neighborhood, and the agent could have `backtracked` to its initial location (and, thus initial signal strength) if it were unable to find a better configuration.

3.2.2 BE-Rebid

The BE-Backtrack approach allows the agents to return to a previously explored state. The SE-Optimistic and SE-Mean approaches allow agents to rebid after every round. Prior work in different decision making contexts [14] has shown that such reevaluation at each timestep can lead to better performance in practice. *Balanced Exploration with Backtrack and Rebidding* (BE-Rebid) combines both of the above algorithms, representing a type of hybrid approach. The agents calculate their gain using the same equations as in BE-Backtrack, but all agents re-calculate and rebid their most favorable action in each time step.

Equation 1 calculates the expected gain of exploring for t_e steps, but if agents rebid on each round, the number of exploratory steps an agent executes may be different from t_e , potentially invalidating the agent’s initial estimate. However, if an agent wins the bid to move and then moves for fewer than t_e rounds, it will be due to signal strengths received after moving: either the moving agent has found itself in a favorable position and no more exploration is needed, or the cumulative signal strength of one of its neighbors has significantly decreased. As an example, consider three agents connected in a chain, as in Figure 2. Suppose that agents 1 and 3 explore and the signal strength between 1–2 and 2–3 drops. It may be more efficient for agent 2 to explore in order to improve both of its signal strengths, but this on-the-fly reasoning is disallowed in BE-Backtrack when $t_e > 1$.

Since the agent is allowed to `backtrack`, the reward enjoyed by the agent after it `backtracks` depends on the maximum reward that it encounters during the exploration phase. In the next section, we consider a similar algorithm, but assume that the agent cannot `backtrack`, removing the ability for an agent to retrace its steps back to a previous location. The expected utility of moving to a new location is given by the mean μ of the distribution, and the immediate reward is known to the agent.

3.2.3 BE-Stay

The *Balanced Exploration with Stay* algorithm applies when agents are unable to backtrack. Based on initial experiments which suggested that BE-Rebid outperformed BE-Backtrack, BE-Stay was designed as another one-step approach where agents make a decision during every round. The intuition behind this approach is that at each round, every agent considers its current total signal strength, R_c , and compares the expected value it would get by staying in the same location (V_{stay}) with the expected value of exploring ($V_{explore}$). Because only one agent in a neighborhood can move at a time, we again assume that no neighbors move when calculating these expected values. The value of exploring can be formulated recursively, where the reward will be zero at time $T = 0$, but the agent can choose to either perform the `stay` or `explore` action at time $T > 0$. The value of exploring is calculated as follows:

$$V(R_c, T) = \begin{cases} V_{stay}(R_c, 0) = V_{explore}(R_c, 0) = 0 & \text{for } T = 0 \\ \max(V_{stay}(R_c, T), V_{explore}(R_c, T)) & \text{for } T > 0 \end{cases} \quad (2)$$

The expected value from stay will be the current signal strength multiplied by the time left in the trial:

$$V_{stay}(R_c, T) = R_c T.$$

The expected value of moving will depend on the probability of achieving a given signal strength in the next state, the reward received for that signal strength on one timestep, and the expected value of the rest of the trial:

$$V_{explore}(R_c, T) = \int_{-\infty}^{\infty} P(x)(V(x, T - 1) + x)dx$$

where $P(x)$ is the probability of receiving the total signal strength x in an unexplored location.

In each round, agents calculate V_{stay} and $V_{explore}$ using Equation 2. If `explore` has the higher expected value, an agent will bid to move for one round, and the agent with the highest bid in each neighborhood will move. Note that BE-Stay differs from BE-Rebid even when the backtrack state is the current state: BE-Rebid assumes the agent may backtrack to this state in the future, which BE-Stay does not.

4 Experimental Results

This section of the paper discusses the empirical validation of the five novel DCOP algorithms in both simulated and physical agents.

4.1 Experimental Setup

Experiments in this section compare the performance of our DCOP algorithms in multiple settings by measuring the cumulative signal strength achieved. Signal strength values were all non-negative integers (which conforms to the CenGen interface of the physical robots), which allowed our implementations to use summations rather than integrations.

In simulation, every experiment is run for 30 independent trials with each of the five algorithms (and one additional time for MGM-Omniscient). Each of 30 trials begin with the same initial configuration to reduce the impact of randomness. In each experiment, lower and upper bounds are determined by disallowing all sensor movement and using MGM-Omniscient, respectively. Results are reported as a scaled gain, scaled uniformly between 0 and 1. Any gain greater than zero represents an improvement directly due to the controlling DCOP algorithm. Such a metric helps isolate the improvement due to sensor movement and scales across tasks with different numbers of links, agents, and horizons. Signal strengths are drawn from a normal distribution with a mean of 100 and a standard deviation of 16.⁵

When experimenting with the physical robots, we do not compare with MGM-Omniscient because the reward matrix is unknown in the real world and instead report the absolute gain. Experiments are conducted with three Create robots running the DCOP algorithm and a fixed radio controller, forming the set of agents in the DCOP. Due to the wavelength used by our robots (5 GHz corresponding to wireless 802.11a specifications), we discretize the state space so that agents move by 2.5cm at a time, $(\frac{1}{2}\lambda)$.

On each round, robots wait one second for signals to stabilize and then calculate their signal strengths by querying the CenGen interface. 10 samples, collected $\frac{1}{4}$ second apart, are averaged to define the agent’s current signal strength. Robots were placed at a distance of 5–10 meters apart in a non-line of sight configuration. Since the objective of the agents was to maximize the cumulative reward in the given time horizon, we did not analyze the runtime of the experiments either in simulation or on physical robots.

Results in simulation (Section 4.2) show that the performance of the hybrid BE-Rebid approach is statistically the same, or better than, all the other algorithms (except for very short experiments). Results on hardware (Section 4.3) show that these algorithms are able to provide significantly improvements over no optimization. Additionally experiments show that the balanced exploration techniques work better than SE-Mean and SE-Optimistic for the majority of settings.

4.2 Simulation Results

This section presents three sets of results, each of which varies a different component of the problem domain: the number of agents, the time horizon, or the network topology. First, however, consider the learning curve in Figure 3, which shows the total reward per timestep of our five algorithms, along with MGM-Omniscient (topmost) and NoMovement (bottom most). This learning curve depicts the algorithms’ rewards over time. For each algorithm, the total cumulative signal strength will be the area under the curve, and the gain will be the area between the curve and the NoMovement line. The y -axis shows the total signal strength and the x -axis shows the time horizon. SE-Mean converges quickly to a comparatively low value while SE-Optimistic continually explores, attempting to achieve the maximal signal strength. BE-Stay can not backtrack and thus must be more cautious; it converges the fastest of all three BE methods. BE-Backtrack attains the highest final reward, but it takes much longer to converge than BE-Rebid and does not achieve the highest cumulative signal strength (BE-Rebid explored for only 36 rounds and received a final reward of 13,198, whereas BE-Backtrack explored for 76 rounds and received a final reward of 13,347; the cumulative rewards for BE-Rebid and BE-Backtrack were 666,062 and 659,925 respectively).

⁵The range $\mu - 6\sigma$ to $\mu + 6\sigma$ covers 99.999% of the samples for a normal distribution. We therefore considered signals within the range [0,200].

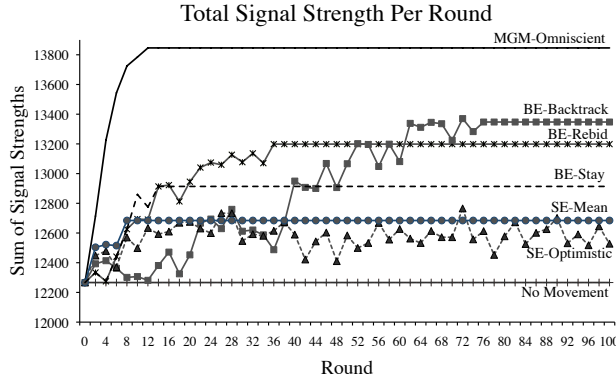


Figure 3: A representative learning curve for 20 agents in a chain topology where $T = 100$.

Having examined a single trial, consider the first set of results in Figure 4(a) which examine how the algorithms' relative performance changes as the number of rounds is increased. The y -axis measures the scaled gain, where $y = 0$ is equivalent to no sensor movement and $y = 1$ on the y -axis is the cumulative reward from MGM-Omniscient. The x -axis shows the five values of T , the total number of rounds in a trial, used in the experiment. All trials used random sparse graphs with 15–20 links and 10 agents. Each result is averaged over 30 independent trials and the error bars show the standard error. The difference between scaled gain for each pair of algorithms are statistically significant within a single value of T (paired Student's t -tests calculate $p < 0.05$), except for $T = 5$. When the time horizon is very small, SE-Mean and the BE algorithms all performed roughly the same because all four algorithms performed very little exploration. As the number of rounds per experiment increases, BE algorithms outperform SE algorithms, and BE-Rebid consistently achieves the highest scaled gain.

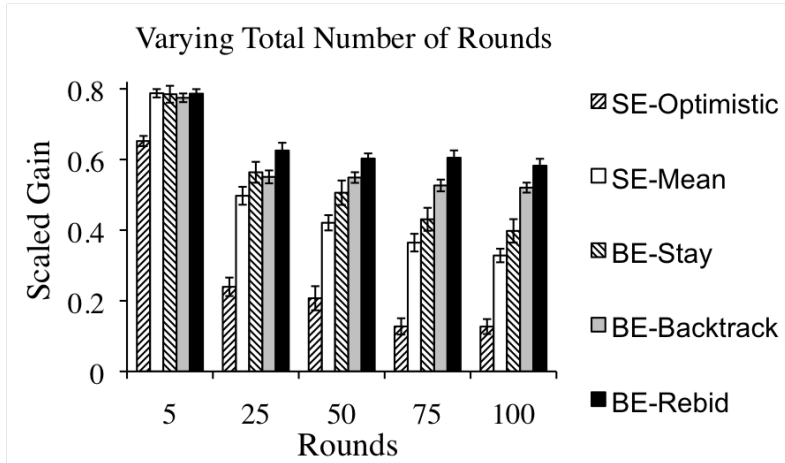
The second set of experiments, summarized in Figure 4(b), varies the number of agents. The y -axis again depicts the normalized gain. The x axis shows the number of agents, varied from 5 to 50. Paired Student's t -tests determine that the results are statistically significantly different ($p < 0.05$) within all sets of tests of the same number of agents. The performance of BE-Rebid was significantly better than the performance of other algorithms in all cases.

The third set of results shown in Figure 4(c) compares the performance on different graph topologies: a chain structure, random structures (with $\frac{1}{3}$ or $\frac{2}{3}$ of all possible links enabled), and a fully connected topology. Each test uses 20 agents and 100 rounds. All results within a single topology are again statistically different ($p < 0.05$). The y -axis again measures scaled gain and error bars show standard error. Results are again averaged over 30 trials, all with random payoff matrices. All results within a single topology are statistically different ($p < 0.05$).

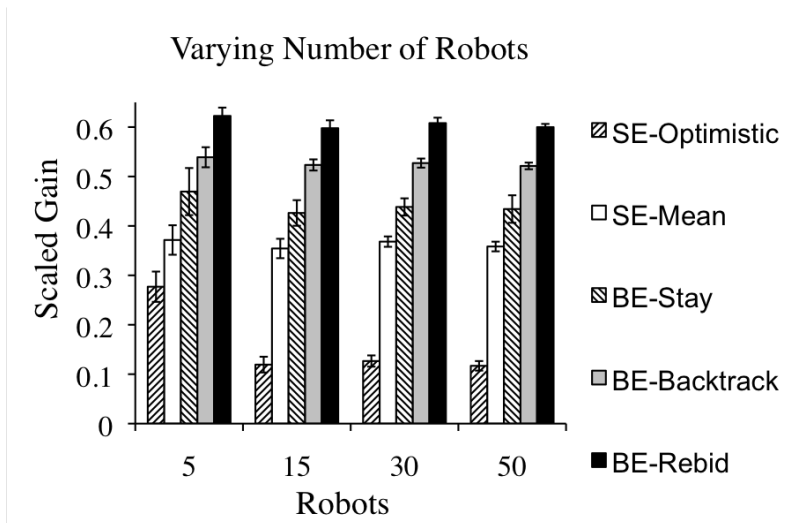
Three trends in Figure 4(c) are worth noting. First, BE-Rebid statistically significantly ($p < 0.05$) outperforms all other algorithms in all topologies tested, except in the fully connected graph (where it is roughly equivalent to SE-Optimistic as only one agent can move per round). Fully connected graphs are thus one setting where the relatively simple static estimation algorithms can perform just as well as the more complex BE algorithms.

Second, as the link density of the graph is increased, the relative performance of BE-Backtrack *decreases*, with statistical significance ($p < 0.05$), due to the aggressive nature of the algorithm. A BE-Backtrack agent will explore for t_e steps, preventing all neighbors from moving during this time. Thus, as the link density increases, higher number of agents are not allowed to move until after t_e steps.

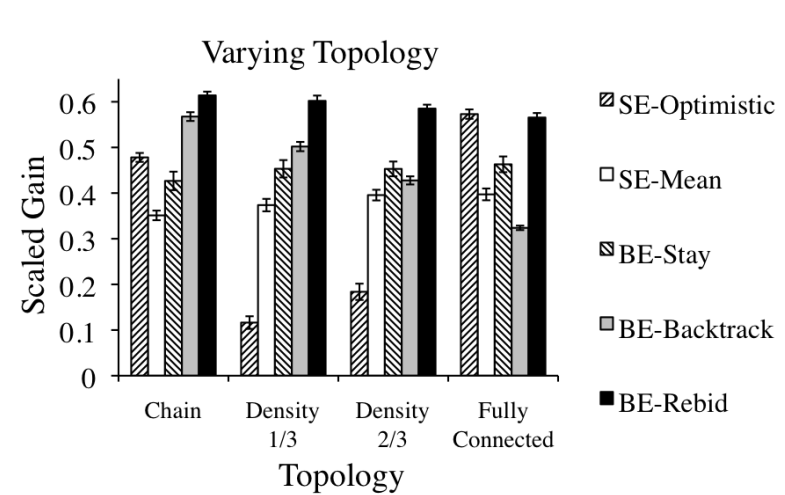
Third, SE-Mean outperforms SE-Optimistic in randomly generated graphs, but not in chain and fully connected graphs. Unlike in chain and fully connected graphs, agents in random graphs can have a high variance in their degrees of network connectivity. We analyze the number of agents that were able to optimize their rewards. While 40% of the robots moved when running SE-Mean, only 18.5% robots could do so when running SE-Optimistic in random graphs with density $\frac{1}{3}$. SE-Optimistic agents with a high degree of connectivity monopolize movement opportunities because they bid



(a) The performance of different algorithms when the time horizon is varied



(b) The performance of different algorithms when the number of robots is varied



(c) The performance of different algorithms when the graph topology is varied

Figure 4: The performance of different algorithms is shown where the y -axis is the scaled gain (0 represents No-Movement and 1 represents the gain of MGM-Omniscient), the x -axis describes the setting, and the error bars show standard error.

unrealistically high rewards. Their bid is relatively large when compared to the bids of agents with lower degrees. There exists a large correlation (Pearson’s coefficient of $\rho > 0.5$) between the degree of the agent and the number of moves made by the agent in SE-Optimistic. In contrast, SE-Mean agents allow others to win bids once the reward of an agent reaches the average over all neighbors. There exists only a weak correlation with $\rho < 0.05$ between the degree of the agent and the number of moves made by the agent for SE-Mean, explaining this difference in performance.

The results also show that BE-Stay is statistically significantly dominated by BE-Rebid, demonstrating that the ability to backtrack can lead to significantly better performance.

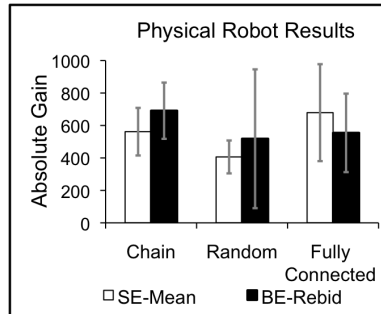


Figure 5: This graph shows the performance of SE-Mean and BE-Rebid for different topologies, as described in Section 4.3. Each experiment summarizes results from 5 independent trials, the y -axis shows the absolute gain in signal strength over the initial configuration, and in all cases our DCOP methods significantly improve over the initial configuration. Error bars show the standard error.

4.3 Physical Robots Results

The previous section showed that our novel DCOP algorithms were able to significantly improve the accumulated signal strength in a simulated environment. This section takes the evaluation one step further by demonstrating that two of our algorithms significantly improve performance on physical hardware, using measured signal strength data as the reward.

Three topologies were tested: chain, fully connected, and random graphs. In the random topology tests, the robots were randomly placed and the CenGen API automatically defined the neighbors, whereas in the chain and fully connected tests the agents had a fixed set of neighbors over all trials. All tests were conducted with a time horizon of 20 time steps and used four agents (three robots and one fixed radio controller). SE-Mean was chosen because it performed best with a small number of agents in simulation and BE-Rebid was chosen because it almost always outperformed all other algorithms.

Figure 5 shows the results of running BE-Rebid and SE-Mean on the robots. The y -axis shows the gain of the algorithms. Note that the y -axis hasn’t been normalized in this figure as MGM-Omniscient could not be run on the real robots (the actual signal strength cannot be determined *a-priori*). The values are signal strengths in decibels (dB). Each value is averaged over five independent trials. For example, when the topology is chain, the average gain of BE-Rebid is 691 units where as the average gain of SE-Mean is 561. BE-Rebid performed better than SE-Mean in the chain and random graphs, but SE-Mean performed better in the fully connected graph. While too few trials were conducted for statistical significance, it is important to note that in all the cases, there is a significant improvement over the initial configuration in the robots. Additionally, because decibels are a log-scale metric, the gains are *even more significant than one may think on first glance*.

Taken as a whole, experiments in this paper show that our DCOP algorithms are able to significantly improve mobile sensor networks in both simulation and hardware. They also demonstrate the superiority of the hybrid approach of BE-Rebid in most situations, except for fully connected graphs or when very few movements are allowed. In such cases, BE-Rebid doesn’t have any additional advantage over simpler approaches like SE-Optimistic.

5 Related Work and Conclusions

Related work in DCOPs has been discussed in earlier sections. Specifically, previous work in distributed constraint reasoning in sensor networks [8, 20] does not use a DCOP formulation or handle unknown reward matrices. A number of other works on mobile sensor networks for communications (c.f., Cheng et al [3] and Marden et al. [11]) are based on other techniques (e.g., swarm intelligence, potential games, or other robotic approaches). Instead, we extended DCOPs as they can scale to large tasks using local interactions. *Reinforcement learning* [17], a popular approach in multiagent learning, does not directly apply in this domain as agents must quickly discover good variable settings, not a control policy. *Optimal Stopping Problems* (c.f., the *Secretary Problem* [6]) optimize the final rank of the selected instance, not on-line metrics, and are exclusively single agent.

This paper focuses on a class of problems that DCOPs could not address before. Apart from early work on distributed constraint reasoning by Lesser et al. [8], this is the first application of DCOPs on physical robots with a demonstrated improvement in performance in a real world problem. We show that such real world domains raise new challenges: (1) agents do not know the initial payoff matrices, (2) the goal is to maximize the total reward instead of the final reward, and (3) agents have insufficient time to fully explore the environment. These challenges open up a new area for DCOP research, as current DCOP algorithms cannot be directly applied. We present and empirically compare five novel DCOP algorithms addressing these challenges. We also present results from two algorithms implemented on physical robots. Our results show significant improvement in the reward in mobile sensor networks. Our experiments demonstrate the superiority of decision theoretic approaches, but static estimation strategies perform well on fully connected graphs or when task time horizon is small. In the future, we anticipate scaling up our evaluation to include additional robots, verifying our algorithms in other domains, and examining alternate reward metrics, such as minimizing battery drain.

References

- [1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38, December 2002.
- [2] R. W. Beard, T. W. McLain, D. B. Nelson, D. Kingston, and D. Johanson. Decentralized cooperative aerial surveillance using fixed-wing miniature UAVs. *Proceedings of the IEEE*, 94(7), 2006.
- [3] J. Cheng, W. Cheng, and R. Nagpal. Robust and self-repairing formation control for swarms of mobile agents. *AAAI*, 2005.
- [4] J. Cox, E. Durfee, and T. Bartold. A distributed framework for solving the multiagent plan coordination problem. In *AAMAS*, 2005.
- [5] S. Fitzpatrick and L. Meertens. Distributed coordination through anarchic optimization. In V. Lesser, C. L. Ortiz, and M. Tambe, editors, *Distributed Sensor Networks*. Kluwer Academic Publishers, 2003.
- [6] P. R. Freeman. The secretary problem and its extensions: A review. *International Statistical Review*, 51, 1983.
- [7] S. Kozono. Received signal-level characteristics in a wide-band mobile radio channel. In *IEEE Transactions on Vehicular Technology*, 1994.
- [8] V. Lesser, C. Ortiz, and M. Tambe. *Distributed sensor nets: A multiagent perspective*. Kluwer Academic Publishers, 2003.
- [9] R. T. Maheswaran, J. P. Pearce, and M. Tambe. Distributed algorithms for DCOP: A graphical-game-based approach. In *PDCS*, 2004.
- [10] R. Mailler and V. Lesser. Solving distributed constraint optimization problems using cooperative mediation. In *AAMAS*, 2004.
- [11] J. Marden, G. Arslan, and J. Shamma. Connections between cooperative control and potential games illustrated on the consensus problem. In *European Control Conference*, 2007.
- [12] P. J. Modi, W. Shen, M. Tambe, and M. Yokoo. ADOPT: Asynchronous distributed constraint optimization with quality guarantees. *Artificial Intelligence*, 161:149–180, 2005.

- [13] A. F. Molisch. *Wireless Communications*. IEEE Press, 2005.
- [14] R. Montemerlo, G. Gordon, J. Schneider, and S. Thrun. Approximate solutions for partially observable stochastic games with common payoffs. In *AAMAS*, 2004.
- [15] J. Pearce and M. Tambe. Quality guarantees on k-optimal solutions for distributed constraint optimization. In *IJCAI*, 2007.
- [16] A. Petcu and B. Faltings. A scalable method for multiagent constraint optimization. In *IJCAI*, 2005.
- [17] R. S. Sutton and A. G. Barto. *Introduction to Reinforcement Learning*. MIT Press, 1998.
- [18] N. Vlassis, R. Elhorst, and J. R. Kok. Anytime algorithms for multiagent decision making using coordination graphs. In *SMC*, 2004.
- [19] M. Yokoo and K. Hirayama. Distributed breakout algorithm for solving distributed constraint satisfaction and optimization problems. In *ICMAS*, 1996.
- [20] W. Zhang, Z. Xing, G. Wang, and L. Wittenburg. An analysis and application of distributed constraint satisfaction and optimization algorithms in sensor networks. In *AAMAS*, 2003.
- [21] Y. Zhang, J. G. Bellingham, R. E. Davis, and Y. Chao. Optimizing autonomous underwater vehicles' survey for reconstruction of an ocean field that varies in space and time. In *AGS, Fall Meeting*, 2005.