In *Proceedings of the Twenty-first Intenational Joint Conference on Artificial Intelligence (IJCAI),*
Pasadena, CA, July 2009.

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

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

Makoto Yokoo
Kyushu University
Fukuoka 819-0395, Japan
yokoo@is.kyushu-u.ac.jp

Abstract

Buoyed by recent successes in the area of *distributed constraint optimization problems* (DCOPs), this paper addresses challenges faced when applying DCOPs to real-world domains. Three fundamental challenges must be addressed for a class of real-world domains, requiring novel DCOP algorithms. First, agents may not know the payoff matrix and must explore the environment to determine rewards associated with variable settings. Second, agents may need to maximize total accumulated reward rather than instantaneous final reward. Third, limited time horizons disallow exhaustive exploration of the environment. We propose and implement a set of novel algorithms that combine decision-theoretic exploration approaches with DCOP-mandated coordination. In addition to simulation results, we implement these algorithms on robots, deploying DCOPs on a distributed mobile sensor network.

1 Introduction

Distributed constraint optimization problems (DCOPs) [Zhang *et al.*, 2003; Mailler and Lesser, 2004; Modi *et al.*, 2005] are a class of problems where cooperative agents must coordinate to maximize some reward. Examples include multiagent plan coordination [Cox *et al.*, 2005] and sensor networks [Lesser *et al.*, 2003]. Because the utility of an agent's action depends on the actions of others, agents must coordinate their individual actions via local interactions to achieve joint goals. Significant progress has been made in the design and analysis of globally optimal DCOP algorithms [Modi *et al.*, 2005; Petcu and Faltings, 2005; Mailler and Lesser, 2004]. However, given that DCOPs are NP-Hard [Modi *et al.*, 2005], optimal solutions require significant communication and computation overhead, motivating the need for locally optimal algorithms that can scale to much larger tasks [Zhang *et al.*, 2003; Pearce and Tambe, 2007].

Given their ability to handle large-scale coordination via local interactions and distributed control, DCOPs are ideally suited for a large class of real-world applications. We target the large class of real-world distributed sensor network applications, including autonomous underwater vehicles used for surveying underwater structures [Zhang *et al.*, 2005] and small mobile robots that establish a communication network. Our study reveals that three novel challenges must be addressed in applying DCOPs to these domains. First, agents are unaware of the initial payoff matrix and must explore the environment to determine rewards associated with different variable settings. Second, the agents' objective is to maximize the total accumulated reward rather than the final instantaneous reward. Third, agents face a limited time horizon, necessitating efficient exploration. These challenges disallow direct application of current DCOP algorithms as they implicitly assume knowledge of the full payoff matrix. Furthermore, time constraints disallow using a globally optimal algorithm as agents cannot fully explore their environment.

This paper introduces and analyzes a family of five novel DCOP algorithms to address these challenges, based on both decision theoretic exploration strategies and simpler "static estimation" strategies. These algorithms are based on two key ideas: (1) exploration and exploitation are interleaved within the context of locally optimal DCOP algorithms, and (2) different settings may require different exploration strategies.

We implement our new DCOP algorithms not only on simulated agents, but on physical robots as well. Our empirical domain is based on a real-world problem in which robots maximize the accumulated signal strength in a mobile sensor network within a time limit. While we acknowledge work on the early precursors to DCOPs in sensor networks [Lesser *et al.*, 2003], our work is the first test of DCOPs on physical distributed hardware and to address the problem of unknown reward matrices. Key experimental results include: (1) in most circumstances, algorithms that update their reasoning after every variable assignment dominate algorithms that pre-compute a multi-step exploration strategy, (2) relatively simple strategies are sufficient in networks that are fully connected and are superior only when the task time horizon is short, and (3) algorithms are able to achieve improvements in reward of up to 80% of an artificially "omniscient" algorithm.

2 Problem Description

2.1 Motivating Domain

This paper focuses on tasks where agents do not initially know their rewards, there is a fixed time horizon, and on-line reward is critical. One such task is *wireless sensor networks*,

where the common goal is to monitor a region of the world and report when interesting objects or events are perceived. We assume robots have limited movement abilities such that they may attempt to optimize the wireless connectivity with their *neighbors* but not modify the topology of the network. Each robot knows its *neighbors*, the robots with which it can directly communicate. During natural disasters, rescue personnel may quickly form such a network by placing mobile robots around a disaster site to relay information about endangered victims, fires, etc. The robots need to *optimize the signal strengths* over the network to ensure reliable and effective communication for the duration of the task.

Radio communication in wireless sensor networks has a predictable signal strength loss inversely proportional to the square of the distance between transmitter and receiver. In urban or indoor settings, scattering, reflection, and diffraction create a *multi-path* setting. Radio wave interference, i.e. *small scale fading*, results in significant signal strength differences over small distances, the prediction of which is very difficult [Molisch, 2005]. This paper assumes that small scale fading dominates: if a wireless sensor is moved $\geq \frac{1}{2}$ of a wavelength, it will measure a new (uncorrelated) signal strength from each neighbor. Such signals can be modeled as an independent random number drawn from some distribution [Kozono, 1994]. Our initial experiments suggest a Normal distribution, but our algorithms are distribution-independent and other distributions can easily be substituted.

Given a network and duration of an experiment, our goal is to maximize the sum of signal strengths on all network links over this time. Each experiment is discretized into synchronized *rounds*. A round ends after all robots perform the required computation, finish communication, and execute an action. An action can be either "move" or "stay." The length of a round in our distributed setting is dominated by robot movement, which takes longer than either the computation or communication for the algorithms presented.

Figure 1: An iRobot Create and its wireless card.

Experiments in this paper use a set of Creates (mobile robots from iRobot, shown in Figure 1) and a custom-built simulator based on these robots. Each Create has a wireless CenGen radio card, also shown in Figure 1. Robots rely on odometry to localize. Based on the specifications of the agents used, two distinct cases may be valid. An agent may: (1) either `stay`, or `explore` a new location, or (2) additionally execute `backtrack` (returning to a previous location) if odometry errors can be ignored. In our physical implementation, robots are able to `backtrack`; but we also run experiments assuming they cannot.

2.2 Problem Mapping

A DCOP consists of a set V of n variables, $\{x_1, x_2, \ldots, x_n\}$, 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

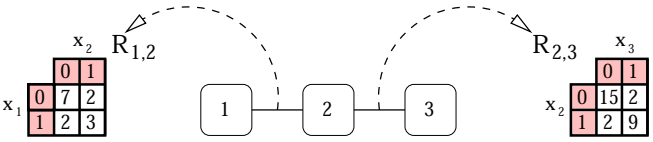

Figure 2: This figure depicts a three agent DCOP.

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 $x_i \leftarrow d_i, x_j \leftarrow d_j \in A$. For example, in figure 2, 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 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.

The agents in a DCOP are traditionally assumed to have *a priori* knowledge of the corresponding reward functions. This assumption may not hold in real-world domains. The problem addressed in this paper involves DCOPs where: (1) agents are not provided reward functions and only know the distributions from which the rewards are sampled, (2) when the agents choose a value, they learn the current rewards on all their constraints, (3) there is a finite horizon T after which the task ends, and (4) the agents' objective is to maximize the cumulative reward over this time horizon T.

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 our novel algorithms, using the mobile sensor network domain as an example. Our algorithms belong to two classes: (1) *static estimation* algorithms which assign a constant estimate to *all* unexplored states, and (2) *balanced exploration* algorithms, which compute the expected utility of exploration based on factors such as time horizon.

Given the inapplicability of globally optimal algorithms, we build on locally optimal DCOP algorithms. The *Maximal Gain Messaging* (MGM) algorithm [Pearce and Tambe, 2007] and DSA [Fitzpatrick and Meertens, 2003] are natural candidates, but DSA has an additional probability parameter that must be set which has a significant impact on its performance [Maheswaran *et al.*, 2004]. While all the algorithms presented are in the framework of MGM, the key ideas can be embedded in any locally optimal DCOP framework. We keep the framework constant to ensure a fair comparison.

MGM-Omniscient: We first implement MGM and artificially provide agents with the reward for each possible value.

Provided such a matrix, MGM-Omniscient will find a locally optimal assignment of values for all agents, and this gives an upper bound. MGM-Omniscient defines a round as a period in which every agent: (1) communicates its current value to all its neighbors, (2) calculates and communicates its *bid* (the maximum gain in its local reward if it is allowed to change values) to all its neighbors, and (3) changes its value (if allowed). An agent is allowed to change its value if its *bid* is larger than all the bids it receives from its neighbors. At quiescence, no one agent can deviate from the proposed assignment and increase the net reward.

3.1 Static Estimation (SE) Algorithms

SE-Optimistic assumes the maximum reward on each constraint for all unexplored values for agents. Thus, in our domain, it assumes that when moving to a new location, the signal strength between it and every neighbor is maximized. On every round, each agent bids its expected gain: *NumberLinks* $\times$ *MaximumReward* $- R_c$ where R_c is the current reward. The algorithm then proceeds as in MGM-Omniscient. This is similar to a 1-step greedy approach where agents with the lowest rewards have the highest bid. Agents typically explore on every round for the entire experiment.

SE-Mean modifies the previous algorithm to assume that visiting an unexplored value will result in the average reward to all neighbors (denoted μ) instead of the maximum. Agents have an expected gain of: *NumberLinks* $\times \mu - R_c$, causing the agents to greedily explore until they achieve the average reward (averaged over all neighbors), allowing them to converge on an assignment. Note that *MaximumReward* and μ can be defined initially as the reward distribution is known.

3.2 Balanced Exploration (BE) Algorithms

These algorithms allow agents to estimate the maximum expected utility of exploration given a time horizon, as well as precisely when to stop exploring within this time horizon. Each agent compares the expected gain it can accrue over the given time horizon from *exploration* against the gain from *backtracking* (or *staying* at current value) for its own bid. This gain from exploration depends on: (1) the number of timesteps left in the trial, (2) the distribution of rewards, and (3) the current reward of the agent, or the best explored reward if the agent can backtrack to a previously explored state. As in MGM, the agent with the highest bid (gain) per neighborhood wins the ability to move.

BE-backtrack takes a multi-step approach, where an agent calculates the expected utility of its current state $V(s)$ given the time horizon T for the task. This expected utility is the maximum of what it would obtain from immediately backtracking to the best known reward for T steps, and the maximum expected utility from exploration. This expected utility is what it uses as its bid to other agents.

BE-Backtrack requires agents to be able to backtrack. It tracks the value with the highest total received reward (R_b). At any point, the agent may return to this value (e.g., a location in our domain) if its neighbors have not changed their values. The state of the agent can thus be defined as (R_b, T): the agent can backtrack to receive reward R_b for T timesteps remaining in the current test. This notion of the agent's state differs from the actual value of the agent's assignment. The utility of backtracking is: $V_{back}(R_b, T) = R_b T$.

If an agent explores, its utility will be based on the reward of the best value found during exploration. 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 value found on round t_e+1, and then `stay` with that value 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 is $t_e \times \mu(n)$, where $\mu(n)$ is the average expected reward over all n neighbors. The second component depends on the probability of finding values with a total reward higher than R_b, multiplied by the number of steps left in the trial. The expected best reward in this case is described by the distribution: $\int_{x>R_b} x \cdot P(x, n, t_e)dx$ where $P(x, n, t_e)$ gives the probability of x being the maximum sample among the t_e samples drawn 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 $\int_{y \le x} f(y, n)dy$. 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 reward greater than R_b, times the number of steps left in the trial. After the agent explores, it will backtrack and receive R_b for the remaining t_s rounds. Again, the cumulative probability of drawing a sample less than or equal to R_b in t_e samples is defined as $F(R_b, n)^{t_e}$, where $F(x, n)$ is defined as before. Thus, $V_{explore}(R_b, T) =$

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

The value of t_e that maximizes $V_{explore}$ gives the number of exploration steps. The expected reward of state (R_b, T) is:

$$V(R_b, T) = \max \left\{ V_{back}(R_b, T), V_{explore}(R_b, T) \right\}.$$

The bid of the agent is $V(R_b, T) - R_c T$ and the agent with the highest bid per neighborhood can change its value (e.g., move) for t_e rounds, after which it backtracks to the value of the highest reward found thus far. BE-Backtrack dictates that the agent will not change its value after backtracking in the $(t_e + 1)^{th}$ round. However, if the agent's neighbors later change their value, 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 reward of an agent that has backtracked after exploring cannot be lower than its reward at the time it started exploring (although it may be lower during exploration). This is because only this agent was allowed to change values in its neighborhood, and the agent could have backtracked to its initial value (and, thus initial reward) if it were unable to find a better configuration.

BE-Rebid agents calculate their gain using the BE-Backtrack equations but all agents re-calculate and rebid at each timestep. Prior work in different decision making contexts [Montemerlo *et al.*, 2004] has shown that such reevaluation can lead to better performance in practice. Equation 1 again calculates the expected gain of exploring for t_e steps, but now the agent may execute fewer than t_e exploratory steps. If this happens, it will be due to rewards received after moving: either the moving agent has found itself in a favorable position and no more exploration is needed or the cumulative reward has decreased significantly and one or more of its neighbors have now won the bid to change their values.

BE-Stay 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 approach where agents make a decision every round.

In this approach, every agent considers its current total reward and compares the expected reward it would receive if it kept the same variable assignment (V_{stay}) with the expected reward of exploring ($V_{explore}$). The reward of exploring, given the current reward R_c, is calculated recursively as:

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

The expected reward from `stay` will be the current reward multiplied by the time left in the trial: $V_{stay}(R_c,T)=R_cT$.

The expected reward of exploring will depend on the probability of achieving a given reward in the next state, the reward received for that one timestep, and the expected reward 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 reward x in an unexplored location.

In each round, agents calculate V_{stay} and $V_{explore}$. If `explore` has the higher expected reward, an agent will bid to change its value. 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

Experiments in this section compare the performance of our DCOP algorithms in multiple settings. In our domain, the number of locations is large, relative to T, and agents are always able to explore new locations. Signal strength values were all non-negative integers which allowed our implementations to use summations rather than integrations. In simulation, 30 random initial configurations were generated. Then each algorithm was run on every configuration. Lower and upper bounds were determined by disallowing all robot movement and using MGM-Omniscient, respectively.

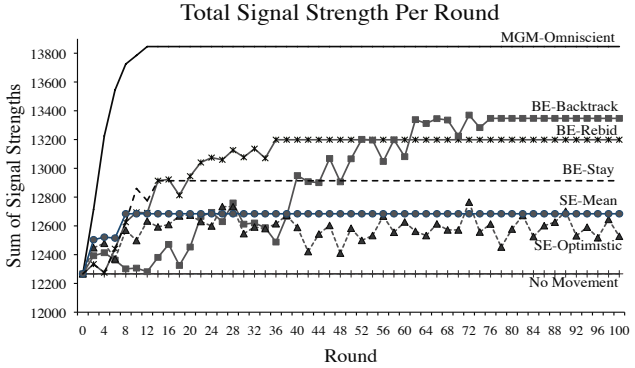

Figure 3: A representative learning curve for 20 agents in a chain topology where $T = 100$.

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 robot movement and scales across tasks with different numbers of links, agents, and horizons. Signal strengths are drawn from a normal distribution defined by $\mathcal{N}(100, 16^2)$. [1]

Experiments with physical robots were conducted with four agents running the DCOP, three of which were Creates and the fourth was a fixed radio controller. Due to the wavelength used by our robots (5 GHz), the state space was discretized so that the agents moved by 2.5cm, or half a wavelength, at a time. In each round, the robots waited for one second for signals to stabilize before measuring their signal strengths (average of 10 samples taken $\frac{1}{4}$ second apart). Robots were placed at a distance of 5–10 meters 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.

4.1 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. However, we first present a single trial to illustrate the typical behavior of our algorithms. Figure 3 is a learning curve from 20 agents in a chain topology. It shows the total reward per timestep when run for 100 rounds each, along with MGM-Omniscient (top line) and NoMovement (bottom line). The y-axis shows the total signal strength and the x-axis shows the time horizon. For each algorithm, the total cumulative signal strength is the area under the curve and the gain is the area between the curve and the NoMovement line. SE-Mean converges quickly to a comparatively low value while SE-Optimistic explores continually, attempting to achieve the maximal signal strength. BE-Stay cannot backtrack and must

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

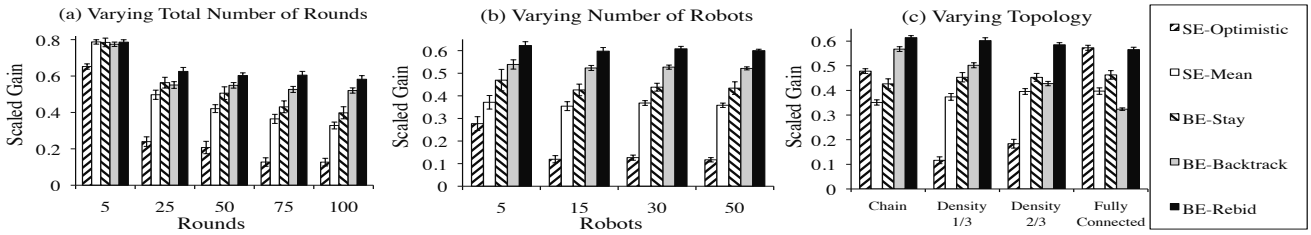

Figure 4: The performance of different algorithms is shown where the y-axis is the scaled gain (0 represents No-Movement and 1 represents the gain of MGM-Omniscient), the x-axis describes the setting, and the error bars show standard error.

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 received 13,347 while exploring for 76 rounds; the cumulative rewards for BE-Rebid and BE-Backtrack were 666,062 and 659,925 respectively).

Having examined a single trial, consider the first set of results shown in Figure 4(a) showing the algorithms' relative performance over different experiment lengths. The y-axis measures the scaled gain. The x-axis shows the five values of T, the total number of rounds in a trial. All trials use 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 is 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, all but SE-Optimistic perform roughly the same because all four algorithms explore very little. 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 results, shown in Figure 4(b), varies the number of agents and again uses random sparse graphs. The time horizon is 100 rounds. The x-axis shows the number of agents, varied from 5 to 50. Paired Student's t-tests determine all results to be statistically significantly different ($p < 0.05$), confirming that BE-Rebid outperforms all other algorithms.

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 ($p < 0.05$).

Three trends in Figure 4(c) are worth noting. First, BE-Rebid statistically significantly ($p < 0.05$) outperforms others 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 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 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.

4.2 Physical Robots Results

Three topologies were tested with physical robots: chain, random, and fully connected. In the random topology tests, the robots were randomly placed and the CenGen API automatically defined the neighbors, whereas the robots had a fixed set of neighbors over all trials in the chain and fully connected tests. Each of the three experiments were repeated 5 times with a time horizon of 20 rounds.

Figure 5 shows the results of running BE-Rebid and SE-Mean on the robots. SE-Mean is chosen because it performed best with a small number of agents in simulation and BE-Rebid is chosen because it almost always outperformed all other algorithms. The gain on the y-axis has not been normalized as MGM-Omniscient cannot be run on the physical robots (the actual signal strength cannot be determined *a-priori*). The values are signal strengths in decibels (dB). BE-Rebid performs better than SE-Mean in the chain and random graphs, but loses in the fully connected graph. While too few trials were conducted for statistical significance, it is important to note that in all cases there is an improvement over the initial configuration of the robots. Additionally, because decibels are a log-scale metric, the gains are *even more significant* than one may think on first glance.

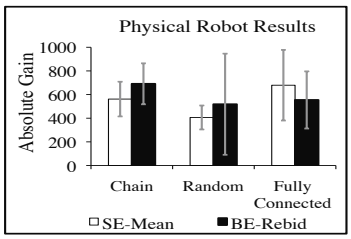

Figure 5: Performance of SE-Mean and BE-Rebid for different topologies show our DCOP methods significantly improve over initial configurations. The error bars show the standard error.

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 [Lesser *et al.*, 2003; Zhang *et al.*, 2003] does not use a DCOP formulation or handle unknown reward matrices. Farinelli et al [2008] also perform decentralized coordination on physical hardware using factor graphs, however, rewards are known and cumulative reward is not considered. A number of other works on mobile sensor networks for communications (c.f., Cheng. et al [2005], Marden et al. [2007]) 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* [Sutton and Barto, 1998], 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* [Freeman, 1983]) 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. [2003], this is the first application of DCOPs that demonstrates 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.

6 Acknowledgements

We would like to thank Rajiv Maheswaran, Chris Kiekintveld, James Pita, Jason Tsai and the reviewers for their helpful comments and suggestions. This work has been sponsored by the DARPA under contract FA8650-08-C-7811 and subcontracted from Lockheed Martin Cooperation.

References

[Cheng *et al.*, 2005] J. Cheng, W. Cheng, and R. Nagpal. Robust and self-repairing formation control for swarms of mobile agents. *AAAI*, 2005.

[Cox *et al.*, 2005] J. Cox, E. Durfee, and T. Bartold. A distributed framework for solving the multiagent plan coordination problem. In *AAMAS*, 2005.

[Farinelli *et al.*, 2008] A. Farinelli, A. Rogers, A. Petcu, and N.R. Jennings. Decentralized coordination of low-power embedded devices using the max-sum algorithm. In *AAMAS*, 2008.

[Fitzpatrick and Meertens, 2003] S. Fitzpatrick and L. Meertens. Distributed coordination through anarchic optimization. In Victor Lesser, Charles L. Ortiz, and Milind Tambe, editors, *Distributed Sensor Networks*. Kluwer Academic Publishers, 2003.

[Freeman, 1983] P. R. Freeman. The secretary problem and its extensions: A review. *International Statistical Review*, 51, 1983.

[Kozono, 1994] S. Kozono. Received signal-level characteristics in a wide-band mobile radio channel. In *IEEE Transactions on Vehicular Technology*, 1994.

[Lesser *et al.*, 2003] V. Lesser, C. Ortiz, and M. Tambe. *Distributed sensor nets: A multiagent perspective*. Kluwer Academic Publishers, 2003.

[Maheswaran *et al.*, 2004] R. T. Maheswaran, J. P. Pearce, and M. Tambe. Distributed algorithms for DCOP: A graphical-game-based approach. In *PDCS*, 2004.

[Mailler and Lesser, 2004] R. Mailler and V. Lesser. Solving distributed constraint optimization problems using cooperative mediation. In *AAMAS*, 2004.

[Marden *et al.*, 2007] J.R. Marden, G. Arslan, and J.S. Shamma. Connections between cooperative control and potential games illustrated on the consensus problem. In *European Control Conference*, 2007.

[Modi *et al.*, 2005] P. J. Modi, W. Shen, M. Tambe, and M. Yokoo. ADOPT: Asynchronous distributed constraint optimization with quality guarantees. *Artificial Intelligence*, 161:149–180, 2005.

[Molisch, 2005] A. F. Molisch. *Wireless Communications*. IEEE Press, 2005.

[Montemerlo *et al.*, 2004] R.E. Montemerlo, G. Gordon, J. Schneider, and S. Thrun. Approximate solutions for partially observable stochastic games with common payoffs. In *AAMAS*, 2004.

[Pearce and Tambe, 2007] J. Pearce and M. Tambe. Quality guarantees on k-optimal solutions for distributed constraint optimization. In *IJCAI*, 2007.

[Petcu and Faltings, 2005] A. Petcu and B. Faltings. A scalable method for multiagent constraint optimization. In *IJCAI*, 2005.

[Sutton and Barto, 1998] R. S. Sutton and A. G. Barto. *Introduction to Reinforcement Learning*. MIT Press, 1998.

[Zhang *et al.*, 2003] 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.

[Zhang *et al.*, 2005] 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.