Applying DCOP_MST to a Team of Mobile Robots with Directional Sensing Abilities

Harel Yedidsion and Roie Zivan,
{yedidsio,zivanr}@bgu.ac.il

Department of Industrial Engineering and Management,
Ben Gurion University,
Israel

Abstract. DCOP_MST is an extension of the DCOP framework for representing and solving dynamic multi-agent applications that include teams of mobile sensing agents. Local search algorithms, enhanced with exploration methods were recently found to produce high quality solutions for DCOP_MST in software simulations. Applying DCOP_MST to robots with directed sensors (e.g., cameras) requires addressing limitations, which were not part of the original design of the DCOP_MST model, e.g., limited angle of the field of vision, collisions between robots etc.
In this paper we contribute to the ongoing effort of applying DCOPs to real world applications by addressing the challenges one faces when applying the DCOP_MST model to a team of mobile sensing robots with directed sensors. We integrate the required adjustments into a new model, DCOP_MST^R, which is the modified version of DCOP_MST for such a real world robot application with directed sensors. The proposed revised model was implemented and evaluated both in software simulations and on a team of robots carrying cameras. Our evaluation of existing algorithms revealed the need to combine actions that change the location of a robot with actions that change its sensing direction in order to achieve effective exploration when solving DCOP_MST^R.

1 Introduction

Some of the most challenging applications of multi-agent systems include teams of mobile sensing agents that are required to acquire information in a given area. Examples include networks of sensors that track targets [19] and rescue teams operating in disaster areas [5]. A crucial, common feature of these applications is that agents select physical locations to move to, and that this selection affects their future interactions, e.g., after moving to some area, a mobile sensor will coordinate its actions with nearby sensors, which it might not have considered before. These types of scenarios have been previously modeled using the Distributed Constraint Optimization Problem (DCOP) framework by representing mobile sensors as agents that need to select locations and their tasks/targets as constraints [11]. Recently, Zivan et al. [19] proposed a model

(DCOP_MST) and corresponding local search algorithms for representing and solving such scenarios, particularly focusing on teams of mobile sensing agents that need to select a deployment for the sensors in order to cover a partially unknown environment. DCOP_MST is an extension of the DCOP model that allows agents to adjust their location in order to adapt to dynamically changing environments. The local distributed search algorithms that were proposed for solving DCOP_MST, were adjustments of standard local search techniques (such as Maximum Gain Message (MGM) [6] and Distributed Stochastic Algorithm (DSA) [17]) to the model, enhanced by specifically designed exploration methods [19]. Nevertheless, prior to this work, the DCOP_MST model and its corresponding algorithms were only tested on a software simulator and not on a mobile robot team.

The task of designing a distributed model for a team of hardware robots is most challenging. Thus, there exists only a handful of studies in which DCOP models were actually used to represent real mobile robots [4,11]. Yet, these studies demonstrate that the community effort of applying the well studied distributed constraint reasoning (DCR) models and algorithms to real world scenarios, must include realistic applications that in many cases reveal challenges and insights that were not faced or discovered in software simulations [4,11].

Thus, our paper advances the research on realistic implementations of distributed constraint models and algorithms by applying the DCOP_MST model to a team of hardware robots carrying cameras. Inspired by the pioneer work of [4] we analyzed the challenges that are raised when shifting to a hardware simulation, adjusted the existing models to represent these challenges, designed an algorithm that copes with the adjustments better than existing algorithms and empirically evaluated the model and algorithm both on hardware and software simulations. Nevertheless, DCOP_MST allows the representation of dynamic elements that were not considered in previous attempts to design representations of mobile sensing agents team scenarios based on the DCOP model, thus, the challenges we face are specific to the assumptions of the model that were made before and do not apply when considering hardware robots with directed sensors. Thus, in this work we adjust the DCOP_MST model to a real world scenario that includes physical robots carrying cameras by:

1. Limiting the field of vision to a directed sight angle. The DCOP_MST model assumed that agents are carrying sensors that can sense all targets within sensing range from the agent's location. This assumption does not hold when using directional sensing equipment, e.g., cameras.
2. In DCOP_MST the sensing ability of an agent (i.e., its *credibility*) was assumed to be uniform for all points within the agent's sensing range. However, the precision of cameras when monitoring targets obviously is affected by the distance from the target.
3. The credibility function of our model take into account uncertainty in actions outcomes and in observations. We account for uncertainty by reducing the credibility as closer the target is to the sensing bounds. When a target is closer to the bound there is

a larger chance that due to imperfect movement or accuracy in observations the target will not be seen. As a result of this function's structure, the agent is encouraged to turn such that the target will be in a direct line of sight.

4. The line of sight between agents and targets must be clear in order for agents to monitor targets using cameras. This is also a requirement that was not part of the DCOP_MST model.

The properties of the real world application mentioned, required an adjustment to the model that affected the representation of the local environment of an agent in the adjusted model including its domain, neighbors set and constraints. Thus, the value assignments of agents include not only its location but the direction in which it points its camera as well. This implies that the domains include beside the selection of different locations, change of directions as well. Moreover, the credibility function must represent the limitations of the cameras we mention above. We aggregate the required adjustments listed above into a modified version of the DCOP_MST model we call DCOP_MST^R [1].

Our results indicate that algorithms that performed best when solving standard DCOP_MST problems are not as effective for solving DCOP_MST^R scenarios. Therefore, we designed an algorithm with enhanced exploration abilities, inspired by the *Periodic Double Mobility Range* (PDMR) exploration method proposed by [19]. This algorithm outperformed existing algorithms both in our hardware implementations and in software simulations of DCOP_MST^R.

The rest of the paper is organized as follows: Section 2 discusses related work. Section 3 provides the necessary background on the DCOP_MST model. Section 4 presents the adjusted model of DCOP_MST^R and analyzes modeling and algorithmic changes. Section 5 describes the experiments performed to test DCOP_MST^R both on the software simulator and on the robot team.section 6 presents our conclusions.

2 Related Work

DCOP is a general model for distributed problem solving that has been widely used to coordinate the activities of cooperative agents [6,17,13,9]. The DCOP literature offers a rich wealth of solution techniques, ranging from complete approaches [7,8,10,3], which are guaranteed to find the optimal solution, to heuristic methods [17,18,9,14] that do not provide optimality guarantees but can provide high quality solutions for systems of significant magnitude (in terms of number of agents and constraint density). Since DCOPs are NP-hard, heuristics are typically preferred for practical applications where a solution must be returned within a few seconds. A number of papers proposed the DCOP model for representing and solving coordination problems related to sensor networks [1] and mobile sensor networks [2,9,4]. Stranders at al. [11] focus on a mobile sensor placement problem, where a network of sensors must cooperatively measure a

[1] The superscript R stands for Realistic/Robots

scalar field (e.g. temperature) to minimize the measurement uncertainty. The method proposed in that work includes the construction of a DCOP instance for every selection of a limited path for the sensors to follow and gather the information required. The Max-sum algorithm was used for solving the DCOP and coordinating sensors' movements. However, the empirical evaluation did not include real mobile sensors. Farrinelli et al. [2] tested the Max-sum algorithm on a graph coloring problem using real hardware sensors which were moved manually. Despite the mobility of the agents, the domains remain constant. Jain et al. [4] deal with mobile sensor networks and applied their framework to real mobile robots. They proposed a framework of DCOP with unknown rewards and used DCOP algorithms for finding a deployment of agents that serve as transmitters of communication and thus, optimizing a communication network. In their framework, agents are mobile but the domains and the neighbor sets are constant. Zivan et al. [19] proposed the DCOP_MST model for representing DCOP problems with dynamic domains, constraints and neighbor sets, along with adjusted local search algorithms. In [16], the Max-sum algorithm was applied to the DCOP_MST model. Despite the fact that the DCOP_MST model and its corresponding algorithms are geared towards representing and solving real world distributed mobile sensor applications, prior to this work the DCOP_MST model was only tested in software simulations. Thus, the novelty of our work is in analyzing the limitations of hardware implementations of the model and adressing them via an adjusted model and an enhanced exploration local search algorithm.

3 Background

In this section we provide background regarding the problems we aim to solve and the models and algorithms that our work stems from.

3.1 Mobile Sensor Teams

In this subsection we outline general properties of MST modeling as formalized in [19]. We later indicate which assumptions were modified and how. In a mobile sensor team, agents $\mathcal{A} = \{A_1, A_2, \ldots, A_n\}$ are physically situated in the environment, which is modeled as a metric space with a distance function d. We assume that the locations that can be occupied by agents are a set of discrete points that form a subset of the total environment, and the current position of agent A_i is denoted by Cur_Pos_i. Moreover, time is discretized into an undeterminate series of time-steps, and the maximum distance that A_i can travel in a single time step is defined by its *mobility range* MR_i. As for perception, agents have limited sensing ranges (SR_i denotes the sensing range of agent A_i), and each agent can only provide information on targets within its sensing range. Moreover, agents may also differ in the quality of their sensing abilities, a property termed their *credibility*. The credibility of agent A_i is denoted by the positive real number $Cred_i$, with higher values indicating better sensing abilities. In [19] the SR is

uniform in every direction, hence it results in a heterogeneous circle. In this paper however, we modify this assumption. The individual credibilities of agents sensing the same target are combined using a *joint credibility function* $F : 2^{\mathcal{A}} \rightarrow \mathbb{R}$, where $2^{\mathcal{A}}$ denotes the power set of $\mathcal{A}$. We require F to be monotonic so that additional sensing agents can only improve the joint credibility. Formally, for two sets $S, S' \subseteq \mathcal{A}$ with $S \subseteq S'$, we require that $F(S) \leq F(S')$. For simplicity, in the remainder of this paper we will use the standard sum function to aggregate agents' credibilities:

$$F_{sum}(S) = \sum_{A_i \in S} Cred_i$$

Targets are represented implicitly by the *environmental requirement* function ER, which maps each point in the environment to a non-negative real number representing the joint credibility required for that point to be adequately sensed. In this representation, targets are the points p with $ER(p) > 0$. A major aspect of the mobile sensing team problem is to explore the environment sufficiently to be aware of the presence of targets.

We say that targets within SR of an agent, are *covered* by the agent and the *remaining coverage requirement* of the target, denoted Cur_Req, is the environmental requirement diminished by the joint credibility of the agents currently covering the target, with a minimum value of 0. Denoting the set of agents within sensing range of a point p by $SR_{(p)} = \{A_i \in \mathcal{A}[d(p, Cur\_Pos_i) \leq SR_i]\}$, this is formalized as:

$$Cur\_Req(p) = \max\{0, ER(p) \ominus F(SR_{(p)})\},$$

where $\ominus : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$ is a binary operator that decreases the environmental requirement by the joint credibility.

The global goal of the agents is to position themselves so to minimize the values of Cur_Req for all targets. In some cases it may be possible to reduce the values of Cur_Req to zero for all targets indicating perfect coverage. However, in other cases this may not be possible (e.g., because of insufficient numbers or quality of agents). In these cases, we aim at minimizing the sum of remaining coverage requirements for all targets. Such a minimization problem is NP-hard [15].

3.2 The DCOP_MST Model

The DCOP_MST model is a dynamic DCOP formulation modeling the mobile sensor team coordination problem [19]. Specifically, each agent A_i controls one variable that represents its position, and the domain of such a variable contains all locations within MR_i of Cur_Pos_i; consequently, as the agent moves from one location to another, the content of its variable's domain changes. A change in the content of a domain of some variable can induce a change in the constraints that include it. This is because only agents that can take a value within sensing range of a target are included in the constraint that calculates the coverage for this target. Hence, the constraint C_p for a

target p, only involves those agents A_i, whose variable's domain includes a location within SR_i of p. Therefore, as the domains change, the constraints change as well. As a consequence, the set of neighbors for each agent changes over time as the agents move. In more detail, in DCOP_MST two agents are neighbors if their sensing areas overlap after they both move as much as possible in a single time step towards each other . This encodes the fact that such two agents might directly influence each other (i.e., by observing the same target in the next time step).

The *local environment* of agent A_i is the joint area within SR_i from all positions within MR_i from Cur_Pos_i, i.e., it includes all targets that the agent can cover after a single move.

3.3 Adjusting Local Search to DCOP_MST

Standard local search algorithms such as DSA and MGM[2] require that agents will be able to compute the best alternative assignment for their variable. In DCOP_MST, the best alternative is a position that allows the coverage of the targets with the highest current coverage requirement in range. Besides the different methods for detecting the best alternative assignment, the major adjustment of the algorithms to the DCOP_MST model is to consider a different domain and a different set of neighbors in each iteration. As mentioned before, a crucial element for solving a DCOP_MST problem is to provide the agents with a mechanism for exploring potentially sub-optimal solutions that might result in increased reward for the team (e.g., because the agents discover a target that was previously not included in their local environment). A successful exploration mechanism must be balanced with the exploitation of the local knowledge. In other words, agents should explore the area for new targets while maintaining coverage of targets that were previously detected. To this end, a number of powerful methods were proposed in [19], and the most successful one is a periodic strategy, named Periodic Incremented Largest Reduction (PILR). The PILR approach allows the agents in some iterations, to select sub-optimal assignments, such as a move that results in an increase of the Cur_Req function up to a constant bound c. Such a strategy is applied periodically, i.e., every k_1 iterations, agents can perform sub-optimal moves for k_2 iterations. The parameter c controls the increase in cost that agents are willing to accept when performing the sub-optimal moves and, by tuning this parameter, we allow agents to search for new targets while preventing them from abandoning the ones they are aware of. The most successful technique according to [19] was the combination of PILR with DSA (DSA_PILR). Another exploration method that was proposed in [19] is the PDMR, which simply allows an agent to consider points within a larger (double) range than their MR for a small number of iterations. This method assumes that a wider range is possible even though the iteration will take longer. Therefore, the agents consider a wider range only in part of the algorithm's iterations, which repeat periodically.

[2] A detailed description of the implementation of these algorithms in the DCOP_MST model can be found at [19].

4 DCOP_MST^R

Applying DCOP_MST to a team of real robots carrying cameras is a challenging task, since several assumptions made in DCOP_MST in order to represent the technology limitations of agents, e.g., sensing and mobility ranges, do not represent adequately the limitations of the hardware at hand. The modification of these assumptions has implications on key features of the model such as the content of the domains, the sets of agents' neighbors and the constraints agents are involved in. These changes also require different considerations from the solving algorithms.

4.1 Modeling Adjustments

The following elements of the DCOP_MST model do not apply when shifting to a realistic scenario with robots carrying cameras:

1. Sensors in DCOP_MST are assumed to have a 360 degree sensing field, while this assumption holds for sensors such as radio receivers, cameras have a limited field of vision, and the direction they are pointed at matters. This is also true in the case of other directed sensors like lasers or robotic arms.
2. The credibility of the agents when monitoring targets within their sensing range is assumed to be uniform, i.e., the information that an agent can provide for each target within its sensing range from its location is of the same quality. In contrast, the effectiveness of cameras declines as the distance from the target grows. In addition, the ability of the agent to provide accurate information also declines as the angle between the direction the camera is pointed at and the line of sight between the agent and the target grows. The closer the target is to the edge of the field of vision, there is a greater chance the agent will lose sight of it due to technical limitations. The model therefore must take this into account and reduce credibility not only according to the distance from the camera but also by the distance of the target from the central line of sight.
3. Sensing in DCOP_MST is not limited due to obstacles. In other words, the sensing quality depended only on the basic sensor quality (i.e. its credibility), and the distance between the agent and the target. However, when cameras are used, the sensing ability is restricted when obstacles are located between the agent and the target, i.e., there is no direct line-of sight. In general, various types of hardware, e.g., cameras, heat sensors, radio frequency sensors etc. differ in the effect that different types of obstacles have on their sensing quality. In the case of cameras, if a target is hidden behind another target or some other obstacle, it is not observable by the sensor.

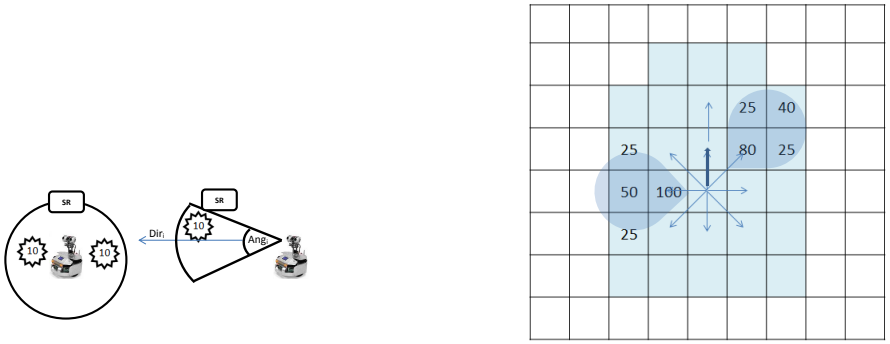
In order to face the discrepancies between the DCOP_MST model and the realistic scenario we discuss above, we propose the DCOP_MST^R $model$, which its details are specified next:

– For each agent A_i, Dir_i is the direction in which its camera is currently pointing at. The directions towards which the sensor can be directed are discretized with respect to the specific problem requirements. The parameter *Number_Of_Directions*, defines the number of possible directions. This parameter can be chosen according to the required accuracy of the specific implementation and the required size of the domain. Larger *Number_Of_Directions* implies greater accuracy but at the cost of increased computation. In our implementation, we have set *Number_Of_Directions* to 8 resulting in eight directions as in a standard compass $(N, NE, E, SE, S, SW, W, NW)$. Figure 1(b) displays the assignments (arrows) included in the domain of an agent with $Dir_i = N$ and *Number_Of_Directions*=*8*.
– The assignment of agent A_i is a pair that includes its position Cur_Pos_i and the direction to which its sensor is pointed Dir_i.
– The domain of agent A_i must include all possible combination of assignments that can result from a single move. Such a move can either be a change of location, or a change of direction in which the sensor is pointed at. In our implementation we assumed that a change of location can only be made in the direction the sensor is pointed at (which was indeed compatible with the limitations of our hardware).
– For each agent A_i, $Ang_i \in (0, 360]$ defines its angle of vision. That is, given its assignment $\langle Cur\_Pos_i, Dir_i \rangle$, agent A_i can monitor a target T only if the distance of target T from Cur_Pos_i is not larger than A_i's sensing range (SR_i) and the angle between the line of sight to T and Dir_i is smaller than or equal to $Ang_i/2$.
– The reduction in credibility resulting from the distance between the agent A_i and target T_j is determined by the *Linear Credibility Reduction Function* ($lin_{i,j}$).
– Similarly, the reduction in credibility as a result of the angle between the direction the sensor is pointed at and the line of sight between agent A_i and target T_j is specified by the *Lateral Credibility Reduction Function* ($lat_{i,j}$).
– Agents sense a target only if there is no other agent or target in the line of sight between them. Each agent A_i holds a boolean variable $c_{i,j}$ for each target T_j in its local environment, which indicates whether the line of sight between them is clear.
– Thus, the effective credibility of agent A_i to target T_j, $Cred_{i,j}$ is calculated by:

$$Cred_{i,j} = \begin{cases} Cred_i \cdot lin_{i,j} \cdot lat_{i,j} \,, & c_{i,j} = true \\ 0 \,, & c_{i,j} = false \end{cases}$$

– Agents are not allowed to be located at the same position. An attempt of an agent to move to a location where another agent is located fails. If two agents move to the same location, only one of them will end up in this location while the other will remain in its previous position (ties are broken randomly).

Figure 1 (a) displays an example of the difference in the sensing capabilities between the two models. The agent on the right represents an agent in DCOP_MST^R. Its sensing range has a limited field of vision represented by Ang_i and a direction Dir_i. In contrast, the agent on the left represents an agent in DCOP_MST with a disc sensing range.



(a) Fields of vision

(b) Domain

Fig. 1: (a) An example of the different fields of vision between DCOP_MST (disc) and DCOP_MST^R (angle and direction). (b) An example of the content of the domain and of the sensing range of a single agent in DCOP_MST^R.

There is a difference in the sensing effectiveness (credibility) between the two models due to the linear reduction. Credibility in DCOP_MST is defined as a step function while in DCOP_MST^R it is defined by a decreasing function In our implementation, $lin_{i,j}$ is calculated in the following way:

$$lin_{i,j} = \begin{cases} d/d_0\,, & 0 < d < d_0 \\ d_0/d\,, & d \geq d_0 \end{cases}$$

Where d is the distance between agent i and target j. We assume that the sensing effectiveness diminishes at a rate of $\frac{1}{d}$ [12]. In addition, the camera cannot recognize the target at zero distance and a minimal distance d_0 is required for the target to be best recognized. We model this distance by a single grid square and set $d_0 = 1$.

Figure 1 (b) displays an example of the sensing range and effectiveness (credibility) of two possible directions an agent in the same location can direct its sensor at, NE and W. The sensing ranges for these directions are represented by the round shaped shaded areas. The quality of the sensing ability reduces with the distance and the angle. The numbers represent the percentage of the full credibility the agent has for each position, i.e. its effective coverage for targets located at these points. These numbers are discretized values derived from the distance and angle credibility reductions. In our implementation, lat is calculated by the following modular decreasing function:

$$lat_{i,j} = (1 - C_{Lat}(Ang_{Dist}/(Ang_i/2)))$$

This function can be fine tuned according to different hardware limitations. Let $Ang_{Dist} = [0, Ang_i/2]$ be the angle between the agent's direction and the line between the agent and the target (Ang_i is the viewing angle of the cameras). $C_{Lat} \in [0, 1]$

is a constant designed to modulate the reduction factor according to the accuracy of the hardware at hand. The larger shaded area in Figure 1 is the agent's *local environment* which contains all of the locations within SR from all assignments in the domain of an agent facing N. The agent would only consider targets located in this area in every iteration.

4.2 Algorithmic Adjustments

Following the adjustments to the model formulation defined in Subsection 4.1, in this section we describe the required adjustments to the algorithms used to solve DCOP_MST^R. We analyze how the model adjustments affect exploration in DCOP_MST^R and propose a local search algorithm with enhanced exploration to overcome these issues.

As mentioned in Section 3.3, among the exploration methods that were proposed for DCOP_MST, the one that was found to be most effective was the Periodic Increment of Largest Reduction (PILR) method, which allows agents to periodically choose a suboptimal assignment, escape local minima and achieve better coverage [19]. Our empirical results indicate that this method does not perform as well when used to solve DCOP_MST^R. We specify several reasons for this difference.

In DCOP_MST agent A_i is only constrained with targets that are located in its *local environment* i.e. the joint area within SR_i from all positions within MR_i from Cur_Pos_i. Any change in the agent's position creates a new *local environment* for the agent. The agent may now take into consideration the coverage requirements of targets that were too far from it to be considered before this last movement. In DCOP_MST^R, an assignment replacement changes in average a much smaller portion of the *local environment* than in DCOP_MST. This is because most of the possible assignment replacements the agent can choose from (i.e., its domain content) result in a change of direction and not a change of location. A direction change results in a much smaller difference in the local environment than a change of location. For this reason, exploration methods that encourage agents to make a single action are sufficient for effective exploration in DCOP_MST but have a minor effect in DCOP_MST^R where the change of locations are more rare.

In addition, in DCOP_MST an agent only has to be within SR from the target to be most effective in covering it, so there are usually several assignments it can move to and still keep covering the same target with the same credibility. This property allows PILR exploration in many cases to keep the same level of coverage while exploring for better assignments. In DCOP_MST^R on the other hand, the agent must be directed towards the target to cover it and any change in position or direction usually leads to a deterioration in the level of coverage.

Another issue that needs to be addressed when designing exploration methods for DCOP_MST^R is the level of exploration for agents that are covering a target which is not right in front of them. The discretized division of possible angles an agent can

direct its sensor towards, often results in a sub-optimal coverage of a close target. This coverage can many time be improved with a combination of actions in which the agent will change location to a position from which it can observe the target directly and adjust its direction accordingly. Obviously, such an improvement in coverage requires more than a single action, thus exploration methods that encourage a single action do not result in such improvements of coverage. In fact, by performing a single move, the agents may decrease the level of their current coverage for the target due to lateral reduction.

The scenarios described above indicate that a method that will allow agents to combine movement and turning actions is required for effective exploration in DCOP_MST^R. Thus, we propose an adjusted version of the PDMR exploration method described in [19] for DCOP_MST^R. In DCOP_MST, PDMR allows agents to periodically consider assignments within twice their MR. In DCOP_MST^R we adjust PDMR so that agents consider a combination of up to two actions (i.e., assignment replacements), e.g., a move forward and a turn assignment or a turn and a move forward assignment. This strategy serves two purposes. It increases the size of the local environment (as PDMR does for DCOP_MST), and it allows agents to improve their coverage by selecting positions from which they can observe targets that are right in front of them.

For agents that are not covering any target PDMR does not result in a sufficient level of exploration since in order to get the same effect as in DCOP_MST, these agents need to travel to locations that are beyond their mobility range, i.e., combine more than one changing location actions. Such a combination has very small probability in DCOP_MST^R, thus, in our adjusted version of the PDMR method we allowed agents that are not covering targets to perform a double combination of actions in each iteration (i.e., they perform DMR and not PDMR).

Following the adjustments to the model that only allow a single agent or target at each location in DCOP_MST^R, we prevent moves that would lead to collisions. From the algorithmic aspect, this adjustment requires an additional message round at the end of each iteration that agent allow neighboring agents to coordinate their movements. If two or more agents aim to move to the same location, the one with the highest improvement in coverage moves and ties are broken according to a predefined order. Agents which are denied movement must reevaluate their future moves and resend their decisions.

5 Experimental Section

In this section we describe the experiments we ran to test the DCOP_MST^R model and algorithms. We conducted tests both on a team of hardware robots and on a software simulation of the model. The tests on the robots were conducted on different scenarios and gave us the insights and understanding of the properties of the model. The software simulation allowed us more flexibility in scaling the number of agents, targets and grid size, and to run many experiments to get statistically significant results.

We used the following parameter values: $C_{Lat} = 1$, *Number_Of_Directions*=*8*, $Ang_v = 60^o$, $MR = 1$ and $SR = 2$. In [19], two scenarios were presented. In the first, two teams of agents, a search team and a surveillance team, cooperated to locate and monitor the targets. In this scenario the target locations are not known and a surveillance agent cannot monitor a target until it is found by a search agent. In the second scenario, only surveillance agents are considered and target locations are assumed to be known. In the experiments described here only the surveillance team was evaluated, i.e., the ER function that the agents used was accurate and was updated after each dynamic event. Due to lack of space we do not include the two-team scenario experimental evaluation. As for the communication model, the agents broadcast their location to the entire network after every movement, but only send messages to their neighbors regarding their next decision.

5.1 Robot Team Simulation

As realistic as we try to make a software simulation, it seldom encompasses the entire scope of challenges that arise in physical applications. As this work is geared towards physical mobile sensor teams, it is imperative that we test our model on hardware robots. To this end, we implemented and tested the proposed model and algorithms on physical robots with the following integral capabilities: movement, communication, responsiveness, vision, localization and decision making. The robots chosen for this system were iRobot Creates mounted with Asus laptops running Ubuntu operating systems and sensing equipment as shown in Figure 2. The sensing equipment consists of PrimeSense cameras mounted on top of the robots. These cameras provide both an RGB map and a distance map. ROS (Robot Operation Systems) has been installed on the laptops. ROS offers a platform for connecting, controlling and programming the robots. The laptops send messages to each other through a router. They receive messages from other neighbors and images from the cameras and process this information according to the algorithm, which they are running. At every iteration, each agent decides on its next move. The targets are simulated by colored cylinders, which are placed in the testing environment and moved manually after several iterations to simulate a dynamic event.

The setting for the hardware experiment included 6 agents. The agents along with 3 targets were randomly located on a 10 by 10 grid. The targets had importance of 100 and the agents had credibility of 60. The algorithms ran for 15 iterations while at the 7th iteration we simulated a target movement. Each reported result in the graphs in Figure 3 is an average over 10 runs of the algorithm on different randomly generated problems.

The results in Figure 3 (a) indicate clearly that DSA_PILR does not have the same level of success as it had when solving standard DCOP_MST [19]. On the other hand DSA_PDMR^R offers a significant improvement over both DSA and DSA_PILR. It is also apparent that following the dynamic event, the proposed algorithm like DSA_PILR and unlike DSA, was able to converge back to solutions of the same quality it found before the dynamic event occurred.

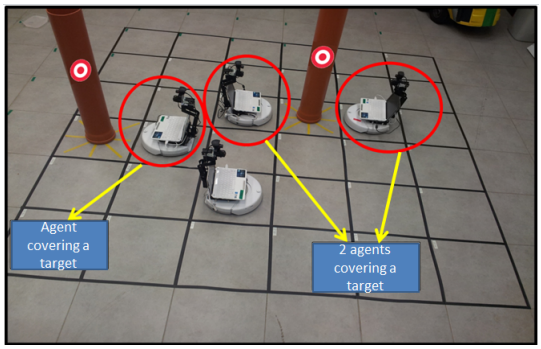


Fig. 2: A team of mobile robots with directional sensing equipment covering targets (orange cons).

(a) Robot team

(b) Software simulator

Fig. 3: A comparison of the algorithms' performance.

5.2 Software Simulation

Software simulations allowed the investigation of larger scenarios that include different sizes of the agent team, a larger number of targets and a larger area. Although, as we mention above, software simulations do not have the same restrictions as hardware implementations, we restricted the agents in the software simulation with the same restrictions that the hardware robots face. The experiments included 10 targets that were randomly located on a 20 by 20 grid. The targets had coverage requirement of 100 and the surveillance agents had credibility of 60. The algorithms ran for 30 iterations while at the 15th iteration we simulated a target movement. Each reported result is an average over 50 runs of the algorithm on different randomly generated problems. The random elements are the initial location of the targets and the agents.

Figure 3 (b) presents results for a mobile sensing team including 20 agents. It is apparent that, like in the hardware experiment, DSA_PILR offers a very small improvement over DSA for DCOP_MST^R. The DSA_PDMR^R version that we proposed in this paper offers a more significant improvement. It is also apparent that the algorithm con-

verges very fast following the dynamic event, to a solution of the same quality as the one it found before the dynamic event. Again, this is consistent with our hardware results.

Notice, that in settings where there are twice as many agents as there are targets, the optimal solution can potentially achieve zero coverage difference (except for some degenerate cases in which some targets are clustered or unreachable). This gives us a good estimation of how well the algorithms approximate the optimal solution. The advantage of DSA_PDMR^R in this scenario grows. Obviously, this increase in the number of agents is exploited better by DSA_PDMR^R than DSA and DSA_PILR.

6 Conclusion

This paper contributes to the ongoing effort of bringing DCOPs to the real world. In this paper we adjust the DCOP_MST model to a realistic robot application where sensors have directional sensing abilities. We adjust the DCOP_MST model for such scenarios by formulating DCOP_MST^R, an extension of DCOP_MST, which includes adjusted elements for which the assumptions made in a simulated environment do not hold. Specifically, we modified the assumption regarding the uniform disc sensing range to represent a directed and limited angle of vision with decreasing effectiveness. Additionally, we incorporated limitations regarding hindered line of sight and collision avoidance. We implemented the top performing algorithms in DCOP_MST, and suggested algorithmic improvements required by the modeling modifications. Interestingly, we find that the top performing algorithm in DCOP_MST, i.e, DSA_PILR, did not perform as well in DCOP_MST^R due to the limited exploration abilities in this model. To this end, we implemented a modified version of DSA_PDMR, which allows for greater exploration by combining movement and direction changes.

As we aim to increase the applicability of DCOPs to real world applications, we implemented the DCOP_MST^R model on a team of hardware mobile robots mounted with cameras. The results of our hardware robot team were consistent with the software simulation results that considered much larger scenarios with teams of various sizes. In all settings the DSA_PDMR^R algorithm we proposed outperformed DSA and DSA_PILR. This improvement was significant both in hardware and software simulations. In all scenarios tested the algorithm was able to reconstruct a solution of the same quality following a dynamic event.

Future work may consider penalize excessive movement and limiting the time to complete iterations.

References

1. A. Farinelli, A. Rogers, and N. Jennings. Agent-based decentralised coordination for sensor networks using the max-sum algorithm. *JAAMAS*, 2013.
2. A. Farinelli, A. Rogers, A. Petcu, and N. R. Jennings. Decentralised coordination of low-power embedded devices using the max-sum algorithm. In *AAMAS*, 2008.

3. A. Gershman, A. Meisels, and R. Zivan. Asynchronous forward-bounding for distributed constraints optimization. In *Proc. ECAI-06*, pages 103–107, August 2006.
4. M. Jain, M. E. Taylor, M. Yokoo, and M. Tambe. Dcops meet the real world: Exploring unknown reward matrices with applications to mobile sensor networks. In *IJCAI*, Pasadena, CA, USA, July 2009.
5. K. S. Macarthur, R. Stranders, S. D. Ramchurn, and N. R. Jennings. A distributed anytime algorithm for dynamic task allocation in multi-agent systems. In *AAAI*, 2011.
6. R. T. Maheswaran, J. P. Pearce, and M. Tambe. Distributed algorithms for dcop: A graphical-game-based approach. In *PDCS*, 2004.
7. P. J. Modi, W. Shen, M. Tambe, and M. Yokoo. Adopt: asynchronous distributed constraints optimizationwith quality guarantees. *Artificial Intelligence*, 2005.
8. A. Petcu and B. Faltings. Ls-dpop: A propagation/local search hybrid for distributed optimization. In *CP-2005*, Sigtes (Barcelona), Spain, 2005.
9. A. Rogers, A. Farinelli, R. Stranders, and N. R. Jennings. Bounded approximate decentralised coordination via the max-sum algorithm. *Artificial Intelligence*, 2011.
10. M. C. Silaghi and M. Yokoo. Nogood based asynchronous distributed optimization (adopt ng). In *AAMAS*, pages 1389–1396, 2006.
11. R. Stranders, A. Farinelli, A. Rogers, and N. R. Jennings. Decentralised coordination of mobile sensors using the max-sum algorithm. In *IJCAI*, 2009.
12. R. Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.
13. M. Vinyals, J. A. Rodríguez-Aguilar, and J. Cerquides. Constructing a unifying theory of dynamic programming dcop algorithms via the generalized distributive law. *AAMAS*, 2011.
14. X. S. W. Yeoh and S. Koenig. Trading off solution quality for faster computation in dcop search algorithms. In *IJCAI*, 2009.
15. G. Wang, G. Cao, P. Berman, and T. F. Laporta. A bidding protocol for deploying mobile sensors. In *in Proceedings of IEEE ICNP*, 2003.
16. H. Yedidsion, R. Zivan, and A. Farinelli. Explorative max-sum for teams of mobile sensing agents. In *International conference on Autonomous Agents and Multi-Agent Systems, AAMAS '14, Paris, France, May 5-9, 2014*, pages 549–556, 2014.
17. W. Zhang, Z. Xing, G. Wang, and L. Wittenburg. Distributed stochastic search and distributed breakout: properties, comparishon and applications to constraints optimization problems in sensor networks. *Artificial Intelligence*, 2005.
18. R. Zivan. Anytime local search for distributed constraint optimization. In *AAAI*, pages 393–398, 2008.
19. R. Zivan, H. Yedidsion, S. Okamoto, R. Glinton, and K. P. Sycara. Distributed constraint optimization for teams of mobile sensing agents. *Autonomous Agents and Multi-Agent Systems*, 29(3):495–536, 2015.