Cooperative Multi-Agent Systems for the Multi-Target κ-Coverage Problem [⋆]

Mirgita Frasheri[1] [0000−0001−7852−4582], Lukas Esterle[2] [0000−0002−0248−1552], and
Alessandro Vittorio Papadopoulos[1] [0000−0002−1364−8127]

[1] Mälardalen University, Västerås, Sweden
{mirgita.frasheri, alessandro.papadopoulos}@mdh.se
[2] Aarhus University, DIGIT, Aarhus, Denmark
lukas.esterle@eng.au.dk

Abstract. When multiple robots are required to collaborate in order to accomplish a specific task, they need to be coordinated in order to operate efficiently. To allow for scalability and robustness, we propose a novel distributed approach performed by autonomous robots based on their willingness to interact with each other. This willingness, based on their individual state, is used to inform a decision process of whether or not to interact with other robots within the environment. We study this new mechanism to form coalitions in the on-line multi-object κ-coverage problem, and evaluate its performance through two sets of experiments, in which we also compare to other methods from the state-of-art. In the first set we focus on scenarios with static and mobile targets, as well as with a different number of targets. Whereas in the second, we carry out an extensive analysis of the best performing methods focusing only on mobile targets, while also considering targets that appear and disappear during the course of the experiments. Results show that the proposed method is able to provide comparable performance to the best methods under study.

Keywords: κ-coverage problem · Collaborative agents · Coalition formation.

1 Introduction

Robots have evolved from performing mundane work to tackling complex tasks with high efficiency. By enabling them to collaborate and interact, they can potentially achieve their tasks faster while requiring less resources. This has reached a level where a group of robots can accomplish tasks that an individual robot would not be able to complete at all. However, collaboration towards a specific goal requires coordination of the all participating robots, and the coordinated formation of coalitions among them. Numerous coalition formation approaches have been proposed which either rely on central components [2, 14, 37] or focus on a single task to be accomplished [33] in order to achieve meaningful interaction and collaboration. These approaches require dissipation of information about available coalitions as well as negotiations about participation of

[⋆] This work was supported by the DPAC research profile funded by KKS (20150022), the FIESTA project funded by KKS, and the UNICORN project funded by VINNOVA.

each potential coalition member [33,37,42]. While coalitions are usually formed around single tasks, the use of multiple teams has been shown to be beneficial when pursuing goals that require multiple tasks to be accomplished concurrently [8, 39]. Moreover, when considering autonomously operating robots that aim to achieve multiple tasks, the individuals have to make decisions on when and how to form coalitions, and to what end the coalition is formed.

In this work, we are interested in the ability of autonomously operating robots to interact and collaborate, without a central component involved, in order to provision varying sets of tasks efficiently. We propose an approach where each robot makes individual decisions about whether or not to provision a specific task. For this decision, each robot employs local information about (i) its own status, e.g. its battery level, (ii) its ability (i.e. having the tools to complete the task), and (iii) its interest (i.e. expected performance value the robot contributes to the collective) in performing such task. More specifically, we propose a novel distributed coalition formation and study this approach in the online multi-object κ-coverage problem [9, 10]. This problem is related to the cooperative multi-robot observation of multiple moving targets (CMOMMT) problem proposed by Parker and Emmons [30]. In contrast, the number of target varies throughout the scenario and targets can appear and disappear randomly. Furthermore, the robots do not have any knowledge about the current number of targets at any time nor the number of robots available to interact. For that reason, the set of robots have to tackle the multiple tasks concurrently.

First, the robots need to discover initially unknown moving objects in the environment. They do not possess any *a priori* information about the number or location of these objects. Furthermore, objects may be mobile, requiring robots to change their own location respectively in order to continuously provision them. This introduces an additional degree of complexity as the robots cannot rely on a single solution but have to continuously re-evaluate their performance and find new optimal solutions on an ongoing basis. Second, each object needs to be provisioned with at least κ robots concurrently, i.e. κ robots having the object within their sensing/actuating region at the same time. Here, detecting new targets is considered the first task, however, every newly discovered target generates a new task of covering this known target for the collective. This generates a trade-off between detecting new objects and covering known objects with κ robots when the collective tries to maximise the duration and number of targets covered by κ robots. However, a robot not only needs to decide between provisioning a specific target or exploring the area to discover new targets, but also which of the different known targets it wants to provision. In order to achieve an efficient outcome in this trade-off, the robots are required to form new coalitions for each individual target. According to the taxonomy of Robin and Lacroix [34], the on-line multi-objective κ-coverage problem is hunting mobile search, monitoring multiple targets, with different viewpoints.

In this paper, we extend previous work that presented a novel distributed coalition formation algorithm considering several tasks [12]. At its core, the proposal consisted in introducing a *willingness to interact* to each individual robot as the main driver for the coalition formation. The *willingness* could depend on a robot's local conditions like battery level, and current level of activity. Utilising this *willingness*, robots can make decisions on whether or not to interact and provision a specific object which eventually leads to forming coalitions with other robots. The experimental evaluation of the aforementioned work has been extended, by investigating additional scenarios with varying values of the κ-coverage, as well as scenarios in which the targets appear and disappear at random times. Additionally, the willingness and utility functions are refined further, and cover for the distance of agents to the targets. The present paper provides an account on both sets of experiments, over several scenarios of increasing number of targets, considering both static and mobile targets, as well as appearing/disappearing targets. The performance is assessed through different metrics, e.g. the average number of agents covering one target, the average coverage time with at least κ agents. Furthermore, the proposed approach is compared against six other methods presented in the literature. The proposed approach shows performance that is comparable to the best methods considered in the experiments.

The remainder of this paper is structured as follows. Section 2 gives a formal definition of the online multi-object κ-coverage problem, whereas Section 3 discusses the related work. Section 4 covers the behaviour of the agents and targets, their interaction as well as our novel coalition formation algorithm. Section 5 gives an overview of the experimental setup, the performed experiments, and the obtained results. Section 6 discusses the generalisation of the proposed approach, while Section 7 concludes the paper and outlines future work.

2 Problem Formulation

In the online multi-object κ-coverage problem, we assume a discrete 2D area Z with a given width and height w and h, respectively, and no obstacles. We also consider a set of active robots $A = \{a_1, a_2, \ldots, a_n\}$, and a set of targets or objects of interest $O = \{o_1, o_2, \ldots, o_m\}$ in this problem. Both robots and objects can freely move within Z, with (nonconstant, yet limited) velocities v_i, where $i = 1, \ldots, n$, and v_j, where $j = 1, \ldots, m$; in their motion the robots will always remain in Z. It is assumed that any robot can move faster than the objects, and that the number of robots and targets is constant. Each robot is controlled by an internal, autonomous software agent. We refer to both as a_i. Each robot has a visibility range, with radius r. An object is covered by any robot, only if it is located within its visibility range. At this point, the robot will determine the number of already provisioning robots for this object. Therefore, it will either initiate a new coalition, in the case of no robots following the target, or join the existing coalition, in the case that less than κ agents are following the target. All objects are appointed to a constant interest level l_j. Moreover, levels of interest can be different for different objects, while also shaping the utility $u_{ij}(t)$ of a robot i for following an object j with interest level l_j at a discrete time-step t.

Every agent i can calculate its willingness w_i to interact with others at each time-step (as detailed in Section 4.3) . This can occur in different situations, e.g. (i) when a robot i first detects an object j entering or leaving its sensing area (ii) when robot i receives a request for help from another robot for following an object j. Thus, the willingness to interact shapes the cooperative behaviour of an agent with respect to other agents, both in terms of asking and giving help. It is assumed that robots are able to change and keep track of their own state and behaviour, as well as the state and behaviour of other robots. Specifically, robot's n state is composed of the following variables: battery level b_i, range d, location $\ell_{x,y}$, and velocity $v_{a,i}$. Without loss of generality, we assume that the level of interest for the targets is equally perceived by all the robots, i.e. the knowledge on the levels of interest for the targets is shared by all agents involved.

The online multi-object κ-assignment problem is solved by having at least κ robots covering any target in the set. Consequently, there are two goals that need to be achieved concurrently: (i) maximising the number of covererd objects, and (ii) covering the targets with at least κ robots. This paper addresses the following questions:

1. What is the average time for which at least κ robots can provision all targets present in Z in an environment when using the proposed coalition formation algorithm?
2. What is the average number of agents that can cover a target with the proposed coalition formation algorithm?
3. How does the motion of the targets affect the obtained performance?
4. How does the defined value for κ affect the performance of the proposed approach?
5. How does allowing for appearing/disappearing targets affect the performance of the proposed approach?
6. How does the proposed method compare with other state-of-the-art techniques for the κ-coverage problem?

We address these questions using two sets of experiments with varying numbers of either mobile or static (immobile) targets, as well as appearing/disappearing targets, according to metrics that analyse the obtainable performance in terms of time to cover targets with at least κ robots, and the average number of robots that cover the targets. Furthermore, we compare our results with six other methods previously proposed in the literature [9].

3 Related Work

The online multi-object κ-coverage problem as introduced by Esterle and Lewis [9] requires moving targets to be detected and afterwards continuously covered by at least κ robots. This is, on one hand a combination of search-and-rescue operations and the κ-coverage problem [18], and on the other hand, an extension to the CMOMMT (Cooperative Multi-robot Observation of Multiple Moving Targets) problem [30].

In search-and-rescue operations, the goal is to coordinate a set of robots to cover a given area and find a set of targets [15]. To achieve this efficiently, Stormont [38] employs a swarming technique to cover a given area rapidly. Bakhshipour et al. [1] employ an heuristic algorithm to optimise and coordinate the movement of individual robots. The combination of supervising robots and subordinate robots for sensing the area enables them to rapidly find all targets and complete the mission successfully. To determine how many searchers are needed for a given operation, Page et al. [29] utilise simulations. They study the trade-off between employed agents and time required to complete a search-and-rescue operation successfully. Yanmaz et al. [41] investigate ad-hoc networks for collaboration in groups of unmanned aerial vehicles and show how these ad-hoc networks allow for more efficient completion of search-and-rescue operations, among others. Ruetten et al. [35] propose using the RSSI of each drone to optimise the area the swarm of robots can cover. This approach is completely distributed and does not require a central component.

Huang and Tseng introduced the idea of κ-coverage to increase the reliability and accuracy in sensor measurements [18]. In addition, the redundancy allows the network nodes to take turns in covering a given area effectively conserving energy resources and increasing the lifetime of the entire network [6, 23, 28]. However, the deployment of the network is calculated *a priori* to ensure optimal coverage of static areas or specific locations. Elhoseny et al. [7] introduced mobile nodes to cover a set of known targets with κ sensors. Using evolutionary computation on a central controller, they find the optimal location for all nodes to maximise the trade-off between number of nodes and coverage of static targets. Liu et al. [26] take one step further and consider a set of unknown targets. This requires their mobile nodes to detect the target points first before optimisation can take place. Utilising Voronoi diagrams, Li and Kao [25] determine the minimum number of required nodes to achieve κ-coverage. Having the Voronoi diagram at hand, they can also calculate the shortest paths for each node to maximise κ-coverage most efficiently. Instead of using a centralised approach, Hafeeda and Bagheri [16] enable the nodes to approximate optimal κ-coverage using a distributed approach. Fusco and Gupta [13] explore the ability of a simple greedy approach to optimally place and orient directed sensors for κ-coverage of static objects in the environment. Micheloni et al. [27] generate activity density maps to identify highly frequented areas. Utilising the density maps, they employ an expectation-maximisation process to determine optimal orientations of Pan-Tilt-Zoom cameras.

The Cooperative Multi-robot Observation of Multiple Moving Targets (CMOMMT) is an NP-hard problem, first introduced by Paker and Emmons [30]. They used artificial force-fields on the set of known targets to attract robots tasked to track them. While they treated all targets equally, Werger and Matarić [40] assigned weights to each target in order to indicate their importance. Each robot broadcasts its eligibility and coordinates directly with others. To learn where targets are located, Jung and Sukhatme [19] learn density maps during runtime and steer the individual robots accordingly. This clusters robots in those areas, leaving the remaining field of operation open and uncovered. To minimise the time where targets are not covered at all, Kollin and Carpin [22] enable each robot to predict when targets will be lost and notify other robots in the area accordingly. This ensures that available robots can move into position and targets can be observed continuously by at least one robot. However, in CMOMMT the main concern is to maximise the overall coverage of all objects with at least one sensor rather than multiple at once.

In their original work [9], Esterle and Lewis enabled agents to advertise detected objects to other agents in the environment. They further studied the effect of different responses basing decisions on aspects like first-come/first-serve, distance, or simple randomness. They further investigated the benefits of utilising topological interactions by learning neighbourhood relations during runtime. King et al. used an entropy based approach attracting agents towards newly detected targets but repelling them when the coverage condition was satisfied [21]. Considering the trade-off between target detection and following targets, Esterle considered the problem as a team formation problem where each team would tackle different tasks, i.e. detecting new targets and following detected ones. Due to the high dynamics of the problem, agents could switch teams during runtime [8].

The interested reader can find more information on κ-coverage, search-and-rescue operations, and CMOMMT [20, 31, 36].

4 Agent Model

In this section, we describe the operation of a robot, in terms of its motion , the manner in which the agent embodied in the robot updates its willingness to interact, and the manner in which the collective of agents makes decisions and cooperates through the proposed interaction protocols. In the following the terms robot and agent are used interchangeably.

4.1 Robot Kinematics

Every robot $a \in A$ follows a simple unicycle kinematic model

$$\begin{cases} \dot{x}_a(t) = v_a(t)\cos(\theta_a(t)) \\ \dot{y}_a(t) = v_a(t)\sin(\theta_a(t)) \\ \dot{\theta}_a(t) = \omega_a(t) \end{cases} \tag{1}$$

where $x_a(t)$ and $y_a(t)$ are the x- and y-coordinate on the map and define the position $\mathrm{p}_a = (x_a, y_a)$ of a robot a at time t, θ_a is the orientation of the robot, v_a is the forward velocity of the robot, and ω_a is its angular velocity. We assume that the robot can localise itself within the map, and that it can detect the obstacles within its visibility range.

A robot $a \in A$ follows a set of objects $O_a \subseteq O$, each of which has a different level of interest l. The direction d_a over which the robot moves is thus computed as

$$\mathrm{d}_a(t) = \frac{\sum_{i \in O_a} l_i(\mathrm{p}_a(t) - \mathrm{p}_i(t))}{\sum_{i \in O_a} l_i} \tag{2}$$

making the robot to move towards all the followed objects, weighted by their respective interest. In this way, the robot will prioritise targets with higher level of interest. The target orientation θ_a° and the forward velocity of the robot are therefore computed as:

$$\theta_a^\circ(t) = \angle \mathrm{d}_a(t), \tag{3}$$

$$\tilde{v}_a(t) = \|\mathrm{d}_a(t)\| \tag{4}$$

where $\angle \mathrm{p} \in [0, 2\pi)$ is the angle of the vector $\mathrm{p} = (p_x, p_y)$ in its reference frame, and it is obtained as $\angle \mathrm{p} = \mathrm{atan2}(p_y, p_x)$. In order to compute the proper value of the angular velocity, we can just use a simple proportional controller with tracking error e_a normalised between $[-\pi, \pi)$:

$$e_a(t) = \theta_a^\circ(t) - \theta_a(t) \tag{5}$$

$$\tilde{\omega}_a(t) = K_p \,\mathrm{atan2}\left(\sin(e_a(t)), \cos(e_a(t))\right) \tag{6}$$

Finally, we include saturations on the forward and angular velocities:

$$v_a(t) = \min(\tilde{v}(t), v_{\max}) \tag{7}$$

$$\omega_a(t) = \min(\max(\tilde{\omega}_a(t), -\omega_{\max}), \omega_{\max}) \tag{8}$$

4.2 Agent Behaviour

Software agents, operate autonomously and their behaviour can be described as a state machine composed of four states: *inspect*, *evaluate*, *inspect & follow*, and *evaluate & follow* (Figure 1). At run-time, all agents start their operation in the state *inspect*, and continue moving in Z according to a given pattern. In case a new target is spotted, or a request is received, an agent switches from *inspect* to *evaluate*. In this state, an agent decides, based on its current state, whether it wants to be involved with the newly detected target or help requests from other agents. The proposed interaction protocol is described in detail in Section 4.4. The result of the interaction is a coalition of agents that will follow the spotted target. If the agent is not part of the coalition after the interaction, it will switch back to the *inspect* state, and it will continue scanning Z for other targets. Otherwise, if the agent is part of the coalition, then it will switch to the *inspect & follow* state. In this new state, the agent follows the target, while it simultaneously inspects for new ones. If all targets an agent has been following go outside its visibility range, then the agent switches to *inspect*. In case an agent either has negative willingness, spots a new target, detects that a target is going outside its visibility region, or it gets a request for help, then it switches to the *evaluate & follow* state. In the latter, the agent either generates a help request, or it responds to a help

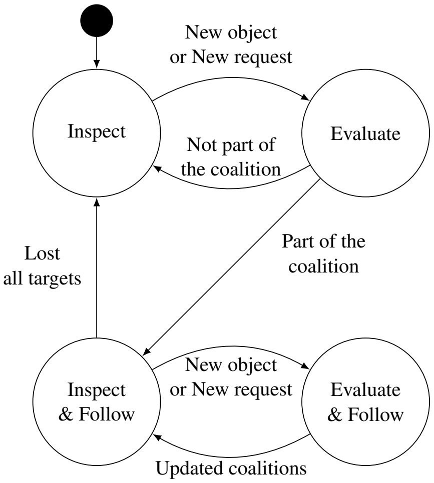

Fig. 1: Agent operation state machine [12].

request. In both cases, the agent decides whether to be part of a (new) coalition, or to drop a target altogether. Once the interaction is complete, an agent switches back to the *inspect & follow* state with an updated set of targets to follow.

Note that an agent can be part of more than one coalition, i.e. can follow several targets at the same time according to their level of interest (as per Eq. 2), but a target is only followed by a single coalition. Also, notice that transitions between states are considered to be instantaneous.

When an agent is following a set of targets, its motion is described by the dynamic model (1), and by the control strategy defined in (3)–(8). The interest level of a target affects the motion of the agent according to (2), i.e. the agent's direction is mostly affected by the level of interest of the targets.

4.3 Willingness to Interact

The willingness to interact w shapes the cooperative behaviour of an agent, i.e. when an agent should ask for help and when it should give help. This parameter reflects the general disposition of any agent to cooperate with the others, without yet considering specific targets. The willingness to interact w takes values in $[-1,1]$. When $w \geq 0$ the agent is able to help other agents in their provisioning tasks. When $w < 0$, the agent will generate requests for help, and for $w = -1$ it cannot continue with the execution of a task on its own. The value of the willingness is updated by each individual agent i based on several individual factors, at discrete time instants t, according to the dynamics:

$$w_i(t+1) = \min(\max(w_i(t) + \mathrm{B}^\top \mathrm{f}(t), -1), 1), \tag{9}$$

$\mathrm{f}(\cdot) = [f_1(\cdot), \ldots, f_m(\cdot)]^\top$ is an $m \times 1$ vector of the m factors that affect the willingness, while $\mathrm{B} = [\beta_1, \ldots, \beta_m]^\top$ is an $m \times 1$ vector that contains the weights of the corresponding factors on the calculation of the willingness.

The calculation of a factor f_i is given by

$$f_i(k) = \phi_i(k) - \phi_{i,\min}, \tag{10}$$

where $\phi(k)$ represents the current measurement of that factor (e.g., the current battery level), while $\phi_{\min}$ is a minimal threshold considered acceptable (e.g., the minimal battery level to perform a task). The terms ϕ and $\phi_{\min}$ take values in $[0,1]$, where 0 is the minimum value of the measured quantity, and 1 its maximum.

In this work, we consider two factors that affect the willingness to interact. These are the battery level b, and the number of objects in O_a currently provisioned by a. Other factors can be included in the calculation of the willingness, without loss of generality of the proposed approach.

Factors can be divided into two categories: necessary and optional. The battery level is a necessary factor, since a robot with a battery level lower than a certain threshold may not be able to achieve its tasks. Therefore, such agents should rely on the help provided by other agents. On the other hand, the number of targets (n_O) an agent is tracking is considered as optional. This is due to the fact that while an agent can follow several targets, its task will become more difficult, e.g. if $1/n_O$ goes below a certain threshold – the agent is following too many targets – then the agent decreases its willingness to give help and consequently increase its willingness to ask for help. The effect of different factors can be modulated by their corresponding weights. The weight for a necessary factor β_{nec} is defined as:

$$\beta_{\text{nec}}(t) = \begin{cases} 1/m, & \phi_{\text{nec}}(t) - \phi_{\text{nec,min}} > 0, \\ -(1+w(t)), & \text{otherwise}, \end{cases} \tag{11}$$

where m is the number of all the factors, whereas the weight for an optional factor β_{opt} is defined as:

$$\beta_{\text{opt}}(t) = \begin{cases} 0, \text{ if } \exists \phi_{\text{nec}}, \phi_{\text{nec}}(t) - \phi_{\text{nec,min}} < 0, \\ \dfrac{\operatorname{sgn}(\phi_{\text{opt}}(t) - \phi_{\text{opt,min}})}{m}, \text{ otherwise}. \end{cases} \tag{12}$$

This ensures that necessary factors have the highest impact on the willingness to interact. As an example, in the case the battery level is below a threshold, then the agent should ask for help, irrespective of other factors ($w = -1$). Thus, the weights of other factors should be set to zero. While we provided an example for factors approaching a minimum, factors approaching a maximum can also be applicable. Nevertheless, the corresponding calculations for the factors and weights have to be adapted accordingly. More examples on factors that can affect the willingness can be found in [11].

4.4 Interaction Protocol

The interaction protocol defines how agents create coalitions for any given target and elect the corresponding leaders for these coalitions. The proposed protocol mostly complies with the Self-organising Coordination Regions (SCR) design pattern [4], however differently from SCR an agent can belong to different coalitions, hence it can have more than one leader. An agent can trigger a help request either when it spots a new target, or it wants to extend an existing coalition to reach $\kappa-$coverage, or it perceives that targets in its visibility range are moving away from itself, or if it is necessary to ask for help (e.g., battery level is under the accepted minimum). Furthermore, agents make decisions on whether to interact with one another when they receive help requests from others. The interaction protocol is illustrated in Figure 2.

When an agent spots a new target, an *information request* is broadcast to other agents, which contains the agent's willingness and respective utility for provisioning the target. Thereafter, the agent waits for a specified time Δt to receive a response from other robots. We assume that agents can identify commonly observed objects and assign common labels. In case a coalition already exists for the given target, the corresponding leader will reply whether more agents are needed to reach the κ-coverage. If no help is needed, then the sending the information request drop the targets and continues its operation. If help is needed, then the agent will receive an assignment from the leader of the coalition, based on the previously sent willingness and utility. In case the agent does not receive a response within time Δt to its initial information request, it assumes there is no other coalition provisioning the target. Subsequently, the process for creating a coalition and electing a leader responsible for following the target is triggered. Initially, the agent calculates its own willingness to help in the future coalition, and the utility

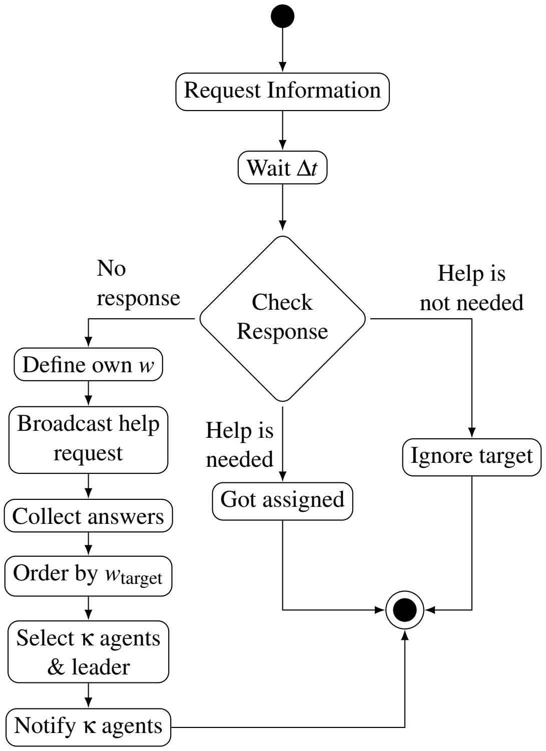

Fig. 2: Activity diagram of the agent's behaviour when a new target is spotted [12].

for following the target. The mechanism follows the logic of a fast bully algorithm [24], well-known in distributed systems. A request for help to follow the object is broadcast to all other agents. Other agents respond with their willingness to help, i.e. the willingness to enter the coalition, and their utility for following the specific target. After the responses are collected, agents with a negative willingness $w < 0$ are discarded. The positive willingness of an agent i to interact is combined with its utility u_{ij} to form the willingness to interact to provision a specific object j at time t:

$$w_{ij}(t) = w_i(t) + u_{ij}(t). \tag{13}$$

Utilities are defined by each agent for the individual target and can generally vary from agent to agent as well as between the different targets. Examples for this could be the size, speed, or direction of movement of the object. In our experiments, we consider different interest levels that are agent-independent, i.e. the agents share the same interest for the same targets. The received values $w_{ij}(t)$ are ordered in decreasing value, and the κ agents with highest $w_{ij}(t)$ are selected for the coalition. The agent with the highest $w_{ij}(t)$ is elected the leader. The outcome is propagated to the other agents via broadcast. The initiating agent does not necessarily need to be part of the coalition.

Furthermore, agents keep track of whether the targets in their visibility range are moving outside the visibility range. We introduce another internal threshold with radius

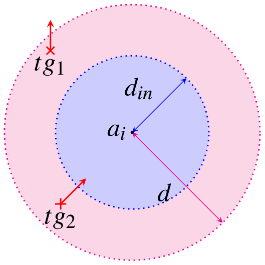

Fig. 3: Agent a_i with visibility range d, indicated with the red circle, and internal range d_{in}, indicated with the blue circle. The targets tg_1 and tg_2 are indicated with crosses, and they are moving towards and away from the agent, respectively [12].

d_{in} around the robots, where $d_{\mathrm{in}} < d$ (Figure 3). When a target, e.g. tg_2 in Figure 3, moves out of the internal range, yet remains within the visibility range, then a request for help is triggered. If a target, e.g. tg_1, is moving towards the agent while being within the internal and visibility range, no request is issued. In case the willingness of an agent becomes negative, help requests are generated, and the agent considers dropping its targets one by one. If the willingness remains negative or becomes -1, eventually all targets will be dropped.

A help request means that either an agent is looking for a replacement for itself, or it is looking for an additional agent that can enter the coalition. This is illustrated in Figure 4. If an agent needs to leave a coalition, we distinguish between leading agents and ordinary members of the coalition. Leader agents should take care of replacing themselves, as such the leader election needs to be repeated. The process can include other agents not yet in the coalition, if κ-coverage is not achieved at that point in time. On the other hand, common agents need only notify the respective leaders when they are dropping a target. Leaders are also responsible for triggering continuously the extension of a coalition in order to reach or maintain κ-coverage, following a monotonically increasing period.

5 Simulation Setup

The behaviour of the agents was evaluated with computer simulations[3], that model the agents' kinematics, behaviour, and interaction. The communication between agents is realised through the robot operating system (ROS) [17, 32]. Two sets of experiments were conducted, namely Set I and Set II. The purpose of the experiments in Set I [4] is to compare the willingness to interact approach against a selection of methods in the state-of-art [9], across several scenarios differing in the number of targets and their mobility – static or dynamic. Whereas, the purpose of the experiments in Set II is to

[3] The code for running the simulations is publicly available at `https://gitagent@bitbucket.org/gitagent/gitagent_2.git`

[4] The present paper is an extension of previous work [12], which contains the account on the experiments and results for Set I.

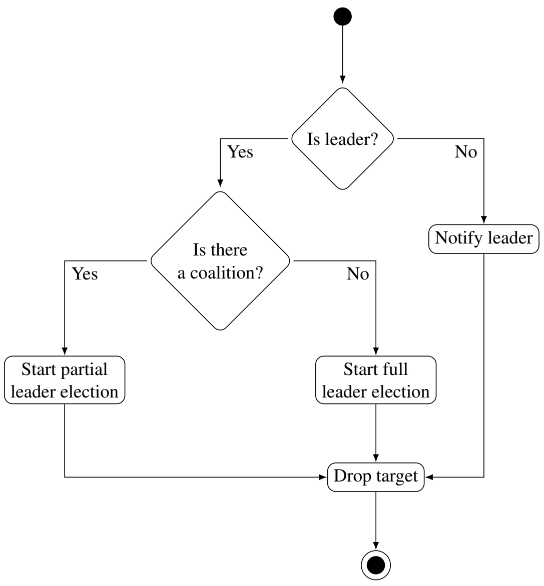

Fig. 4: Activity diagram of the agent's behaviour when it needs to replace itself in a coalition [12].

focus the analysis on the best performing methods identified in Set I, for mobile targets, considering also scenarios in which targets may appear or disappear from the area Z.

5.1 Set I Experiments and Results

In these experiments the method utilising the willingness to interact, was compared to six other methods that were previously proposed in the literature for solving the multi-object κ-coverage problem [9]. Each method is a combination of one communication model and one response model. Two communication models are considered, broadcast *BC* and random *RA*. In *BC* an agent broadcasts a help request to everyone, whereas in *RA* it sends a help request to κ agents chosen randomly. Regarding response models, the following three are considered: (i) newest-nearest *NN*, (ii) available *AV*, and (iii) received calls *RE*. In *NN* an agent will answer to the newest request, and in case of simultaneous requests, it will respond to the nearest one. In *AV* the agent answers to requests according to the newest-nearest strategy only if it is not following other target(s). In *RE*, an agent will answer to requests for objects with the least coverage, only if it is not following other targets. The six methods chosen for comparison, based on these models, are: *BC-NN*, *BC-AV*, *BC-RE*, *RA-NN*, *RA-AV*, and *RA-RE*. Such selection is due to previous results [9], where the broadcast and random communication models were evaluated better with respect to the rest, while the response models were reported to have a significant impact on the κ-coverage.

In all simulations, we consider a total number of $n_A = 10$ robots starting from the same initial position $(0,0)$, with a random direction, and $v_{i,\max} = 2$ units per time-step.

If an agent hits any boundary in Z, it will bounce back at a 90° angle, i.e. Z is assumed to be a limited area surrounded by walls. The objects to be covered are generated uniformly in area Z of size 100m×100m. In our experiments, we consider 7 different scenarios with an increasing number of objects. The number of objects distributed in the environment are 1, 4, 7, 13, 16 and 19 for the corresponding scenarios $S0$ to $S6$. For each simulation the interest level of any target was randomly sampled from a set of levels $\mathcal{L} = \{0.3, 0.6, 0.9\}$. We performed 20 experiments for each scenario, with each experiment having a duration of $T_{\text{sim}} = 300$ discrete time steps and a specified seed. The latter impacts the initial location of the targets, the initial direction for agents and mobile targets, as well as the level of interest of targets for each experiment corresponding to a scenario. Given these settings, we analysed the behaviour of our agents to achieve κ-coverage where $\kappa \geq 3$, and $\kappa \geq 5$.

Results for static targets. In these experiments, targets remain in their initial locations ($v_j = 0$). Once targets are covered, they remain covered for the rest of the simulation, as such, the time for reaching the desired coverage is considered one of the performance indicators for evaluation.

For every scenario, we run N different experiments. For every experiment $e = 1, \dots, N$, we compute for every target j the time to reach 1-coverage $t^{(1)}_{j,e}$, and the time to reach κ-coverage $t^{(\kappa)}_{j,e}$. Based on this information we can calculate: (i) the average time to get one target to be covered by at least κ agents $t^{(\kappa)}_{\text{avg}}$, and (ii) the average minimum time to get all the targets covered by at least κ agents $t^{(\kappa)}_{\text{min}}$. These two metrics give an indication of a minimum coverage, and a complete coverage.. They are formally defined as:

$$t^{(\kappa)}_{\text{avg}} = \frac{1}{N} \sum_{e} \frac{1}{|O|} \sum_{j} t^{(\kappa)}_{j,e} \tag{14}$$

$$t^{(\kappa)}_{\text{min}} = \frac{1}{N} \sum_{e} \max_{j} t^{(\kappa)}_{j,e} \tag{15}$$

In particular, in these experiments we study (i) the average time to get one target to be covered by at least by 1 agent, $t^{(1)}_{\text{avg}}$, (ii) the average time to get one target to be covered by at least by κ agent, $t^{(\kappa)}_{\text{avg}}$, (iii) the average minimum time to get all the targets covered by at least 1 agent, $t^{(1)}_{\text{min}}$, and (iv) the average minimum time to get all the targets covered by at least κ agents, $t^{(\kappa)}_{\text{min}}$. In all the metrics, the lower, the better.

Results for $\kappa \geq 3$ as well as $\kappa \geq 5$ are shown in Figure 5, where $t^{(\kappa)}_{\text{min}}$ is given on the x-axis, and $t^{(\kappa)}_{\text{avg}}$ is given on the y-axis. For both metrics, the lowest the value, the better. In the graph the corresponding Pareto frontier is included in order to highlight the best performing methods. We also compare this directly to the cases for $\kappa \geq 1$ (only a single agent covers the target), however, the agents still aim to cover all targets with $\kappa \in \{3, 5\}$ and therefore might cluster at specific objects even when reporting results for $\kappa \geq 1$.

It can be observed that on average there are no differences between the utilised methods for scenario $S0$ for $\kappa \geq 1$, as shown in Figure 5. This is due to the fact that

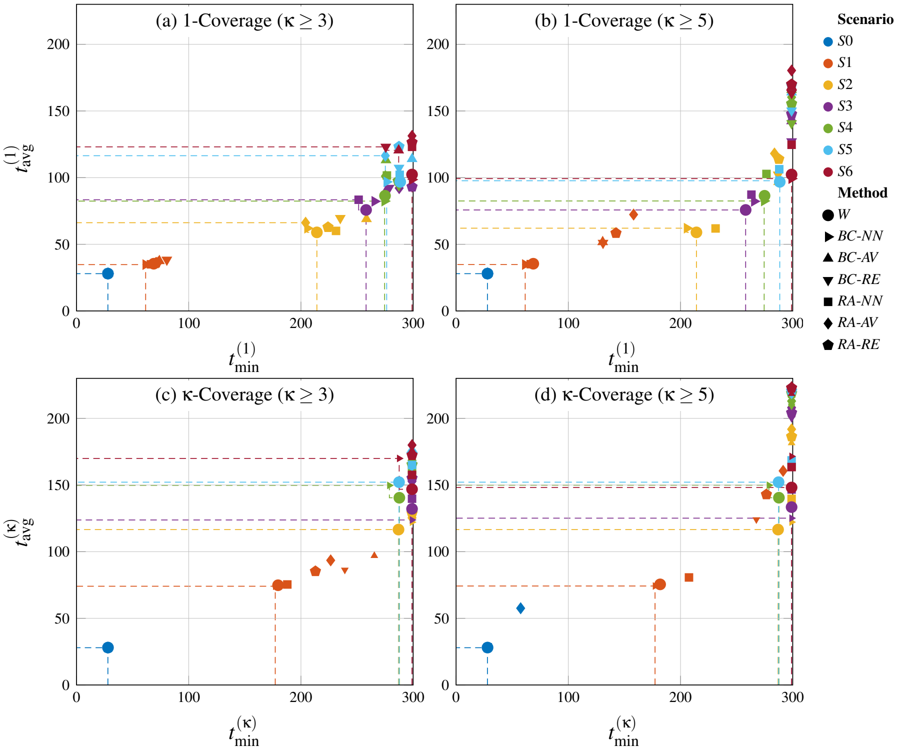

Fig. 5: Average time vs minimum time to cover all stationary targets (i.e., not moving) with 1 (top) or κ agents (bottom), for $\kappa \geq 3$ and $\kappa \geq 5$ [12].

the only static target in the environment will be discovered at the exact same time by any method for an experiment initiated with the same seed. There could be a shift with a couple of time-steps in the discovery times, in case there is an occasional failure or delay in the ROS service calls or broadcast used by agentst when handling targets that appear in the visibility range. Nevertheless, for $\kappa \geq 5$, Figure 5d, the minimum times are not necessarily the same, e.g. the result for method *RA-AV* as compared to the six other methods. With the increase of number of targets in each scenario, the average and minimum times to coverage also increase. For each scenario $S1$–$S6$, a difference on average between the different methods is observed. Mostly, the proposed method, indicated with the 'W' in the legend, is either the best on average or at least on the Pareto frontier, for scenarios $S3$ in Figure 5c; $S2$ and $S5$ in Figure 5b; $S2$, $S5$, and $S6$ in Figure 5d; and for scenarios $S2$ and $S3$ in Figure 5a; $S2$ in Figure 5c; $S4$ and $S6$ in Figure 5b; and $S4$ in Figure 5d, respectively. Similar performance is displayed by the *BC-NN* method, which is the best performing method among the ones considered in this study.

When only one target is involved, the average minimum time to coverage is lowest. In all cases, the agents will move in Z and eventually find and cover the targets.

However, when increasing the number of targets ($S1$–$S6$), agents can gather on the first targets found, leaving remaining targets undiscovered for the rest of the simulation. As such, all metrics are affected, and the $t^{(\kappa)}_{j,e}$ is saturated to the duration of the simulation $T_{\text{sim}} - 1$ [5] for the targets that were not discovered. In Figure 5, the points are accumulated at the $t^{(\kappa)}_{\text{min}} - 1$, which means that in those scenarios not all targets were discovered by the agents.

Results for dynamic targets. In these experiments, targets move within map Z by randomly changing direction, with velocity $v_{t,\text{max}} = 1.5\ m$ per time-step. In both cases, agents move with a higher velocity $v_{a,\text{max}} = 2\ m$ per time-step. Nevertheless, we still use the same sets of scenarios. As for the performance, for a single experiment $e = 1, \dots, N$, we consider the time for which a target j is covered with at least κ agents over the simulation duration, $\tau^{(\kappa)}_{j,e}$, and the average amount of agents that cover the target j over the simulation $\alpha_{j,e}$. Based on these two quantities we compute the following metrics: (i) the average time for which at least κ agents cover the targets, $\tau^{(\kappa)}_{\text{avg}}$, and (ii) the average amount of agents that cover the targets α_{avg}. These quantities are computed as

$$\tau^{(\kappa)}_{\text{avg}} = \frac{1}{N} \sum_{e} \frac{1}{|O|} \sum_{j} \tau^{(\kappa)}_{j,e} \tag{16}$$

$$\alpha_{\text{avg}} = \frac{1}{N} \sum_{e} \frac{1}{|O|} \sum_{j} \alpha_{j,e} \tag{17}$$

While in the experiments featuring static targets, a target will be covered for the whole duration of the simulation once a coalition is formed, in the dynamic case, such assumption cannot be made, because agents can lose targets as all objects are moving in Z. Furthermore, these metrics are calculated twice for *active* and *passive coverage*, i.e. by (i) considering the targets which are actively being followed by agents, and (ii) considering targets that are within the visibility range of agents, but are not being actively followed. In other words, agents are actively following targets when they adjust their own motion based on the motion of the targets.

Results are shown in Figure 6, where the average time of coverage is given along the x-axis, and the average number of agents is given on the y-axis. The method with the willingness indicated with W in the legends of Figure 6 is overall on the Pareto frontier for $\kappa \geq 3$, with an exception for scenario $S3$. With respect to $\kappa \geq 5$, the method with the willingness is on the Pareto frontier for scenarios $S0$–$S4$, and the best on average for $S5$–$S6$. The same is observed in the results for passive provisioning. Furthermore, our approach tends toward maximising the number of agents covering a target, thus it lies on the left side of the Pareto frontier. Similarly to the results in the static case, the performance of the *BC-NN* strategy is comparable to the method with the willingness.

We can observe that for both $\kappa \geq 3$ and $\kappa \geq 5$ the average coverage time is highest when the number of targets is lower, in $S0$ and $S1$, falls for $S2$–6 when the number of targets to be covered increases. We speculate that an increase in the number of targets,

[5] In the final time-step the multi-agent system shuts down, hence this time-step is not considered when dealing with the results.

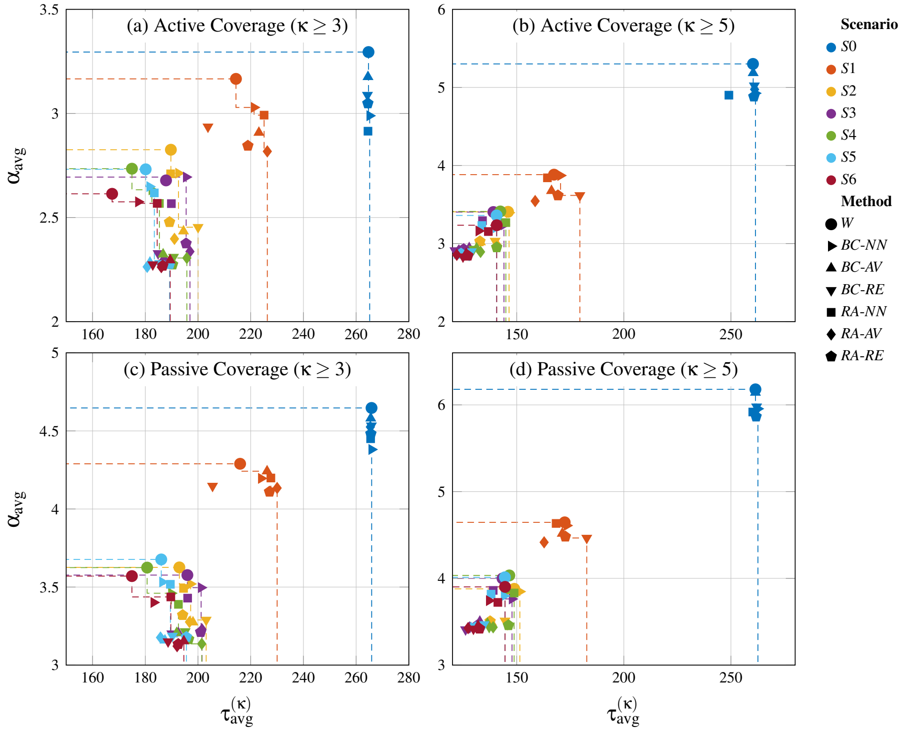

Fig. 6: Average number of agents covering targets vs average coverage time of all targets of the entire duration of the simulation for $\kappa \geq 3$ and for $\kappa \geq 5$. On the top we show active coverage where we only consider the agents actively following a target while the bottom includes passive coverage — agents having multiple targets in their visibility range while following another target [12].

whilst the size of the area is unchanged, might increase the average coverage time as agents can join multiple coalitions, i.e. are able to follow more tasts concurrently. However, this remains subject to further research. The impact of the chosen values for κ can be observed as well in Figure 6, by inspecting the average coverage times, which are lower for $\kappa \geq 5$ than $\kappa \geq 3$. Taking into account what is being covered passively increases the average number of agents that cover a target.

Note that, the averages are taken over all time-steps of the simulation including the time to discover the objects in the first place. Naturally, the lack of coverage prior discovery penalises the shown results.

In our current approach, the agents are not aiming to exceed the desired coverage. Nevertheless, race-conditions in the coalition formation process can result in coalitions that extend the κ-coverage. Furthermore, this can also take place when an agent detects that a target is moving away, and finds a replacement agent which also joins the coalition. Note that, the target is not dropped by the former agent until it actually goes out of its visibility range.

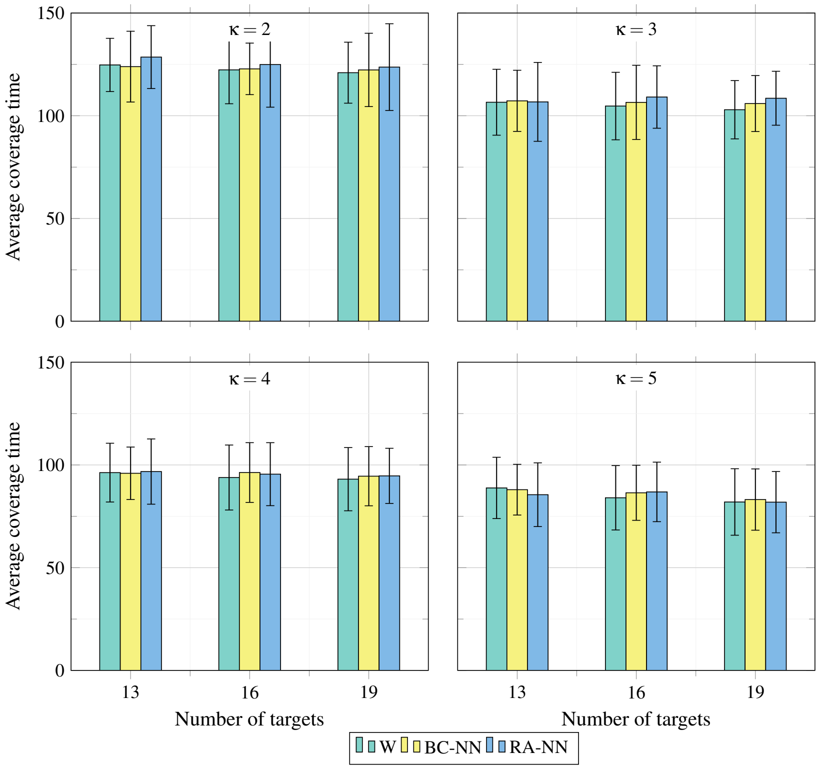

Fig. 7: Average active coverage time for $\kappa \in \{2,3,4,5\}$.

5.2 Set II Experiments and Results

Extension of Set I experiments. Results from the experiments in Set I indicate that the best performing methods across the investigated scenarios are the willingness approach, $BC-NN$, followed by $RA-NN$. In order to gain a better insight on the performance of these methods, additional experiments with different values of κ were run, specifically for $\kappa \in \{2,3,4,5\}$. In these experiments, only mobile targets are considered with the metrics adopted in Set I, and scenarios with number of targets in $\{13,16,19\}$, i.e. scenarios *S4-S6*. Furthermore, instead of running 20 experiments with different seeds, 10 different seeds were used, where for each seed 5 runs were performed, for a total of 50 experiments. The reason behind this choice is that experiments initiated with the same seed do not necessarily produce the same exact results since it cannot be guaranteed that the same coalitions will be formed every-time. Based on which message arrives in time when an agent is gathering responses, a specific coalition will be created. Therefore, affecting the rest of the simulation in terms of which targets are followed or discovered next.

The results of these experiments are given in Figures 7, 8 for active coverage, and Figures 9, 10 for passive coverage. Note that the metrics have been weighed based on

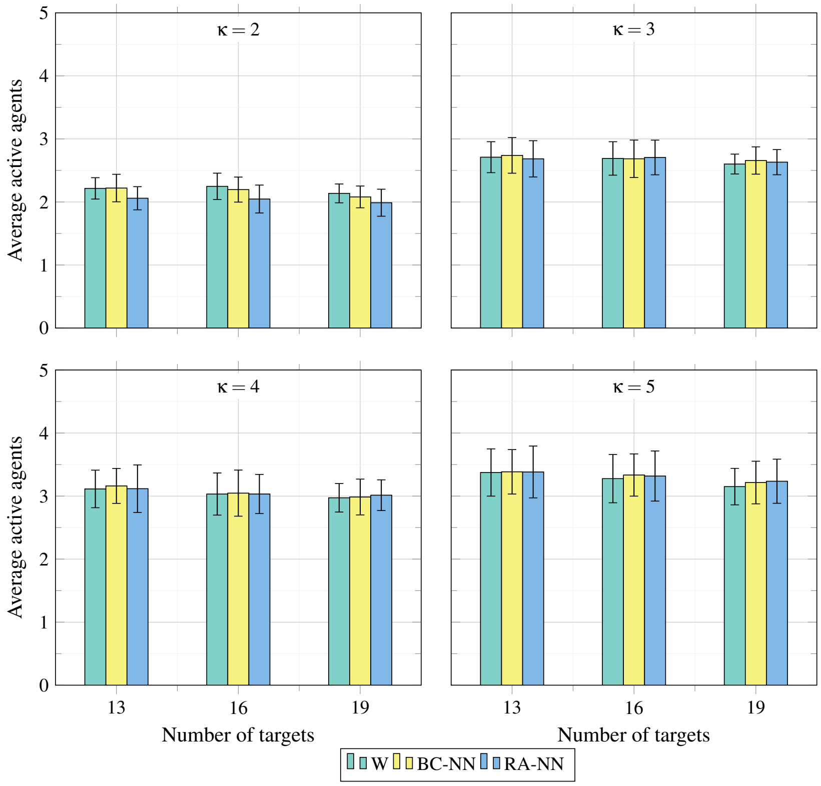

Fig. 8: Average active number of agents per target for $\kappa \in \{2,3,4,5\}$.

the interest level of the targets. It can be observed that as κ increases, the average coverage time decreases (Figure 7), as is expected. The average agents per target metric decreases in the same manner (Figure 8). Additionally, for $\kappa = 2$ the coverage is achieved and surpassed, while in the other cases it is not reached by any method. For $\kappa = 3$ the results are quite close to the desired coverage, however for $\kappa = 4,5$ the achieved coverage is rather below the desired values. Results improve overall when considering passive coverage (Figures 9, 10), and the average agents per target metric lies in $(3,4]$ across the different experiments. Conclusively, the performance of the methods is comparable, with slight differences in average across the different scenarios.

Experiments with appearing/disappearing targets. A more relaxed version of the κ-coverage problem, in which the targets are not only mobile but are also allowed to appear/disappear during the simulation, is used to further evaluate the three methods considered in the previous paragraph. For these experiments, it is assumed that area Z is confined, however it is possible for new targets to come in. The decisions concerning the removal and insertion of targets are made at every time-step. Furthermore, these decisions are sequential, i.e. first it is decided on whether to remove an existing target, thereafter it is decided on whether to insert a new one. The probabilities for removal

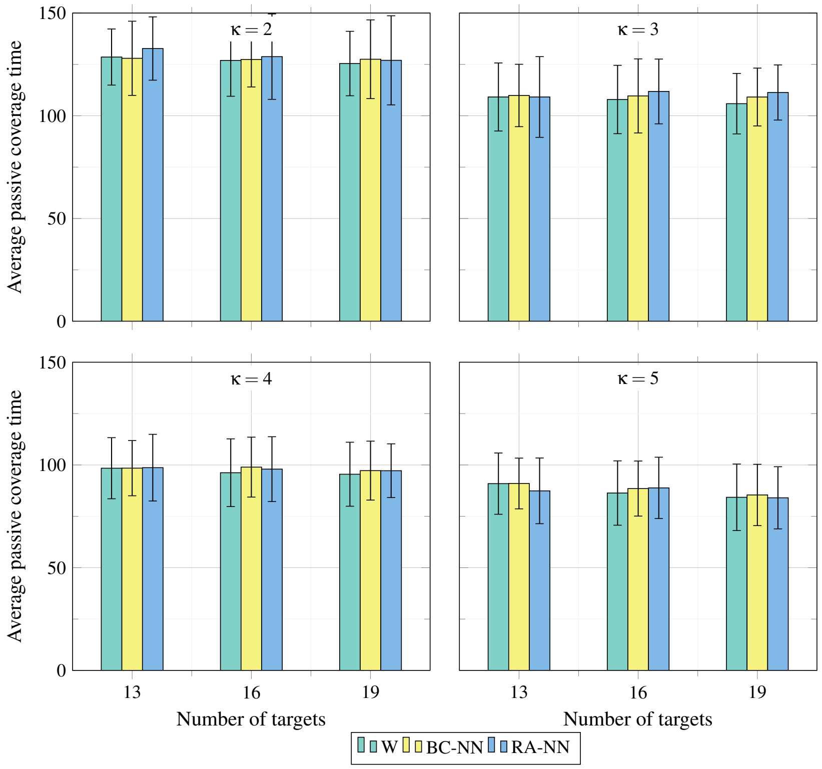

Fig. 9: Average passive coverage time for $\kappa \in \{2,3,4,5\}$.

and insertion are set to the same value $p_R = p_I = 0.2$. Targets appear and disappear on the spot. The rationale behind such implementation is that, a target disappears when there is no interest associated to it, and appears when the interest jumps from 0.0 to $I_j \in I = \{0.3, 0.6, 0.9\}$. Initially, the number of targets present in the simulation is equal to 19, corresponding to the *S6* scenario in the previous experiments. Simulations were run for values of κ in $\{2,3,4,5,6,7,8\}$.

The willingness to interact function has been simplified to account only for the state of the robot at a given time-step, thus Equation 20 is modified as follows:

$$w_i(t+1) = \max(\min(\mathrm{B}^\top \mathrm{f}(t), -1), 1). \tag{18}$$

Furthermore, only one factor shapes the willingness in these experiments, that is the level of activity calculated as:

$$l_A = \frac{O_a}{O_{ALL}}, \tag{19}$$

where O_a is the number of objects provisioned by robot a, whereas O_{ALL} is the number of all objects of interest in Z. Note that, for experiments with appearing/disappearing

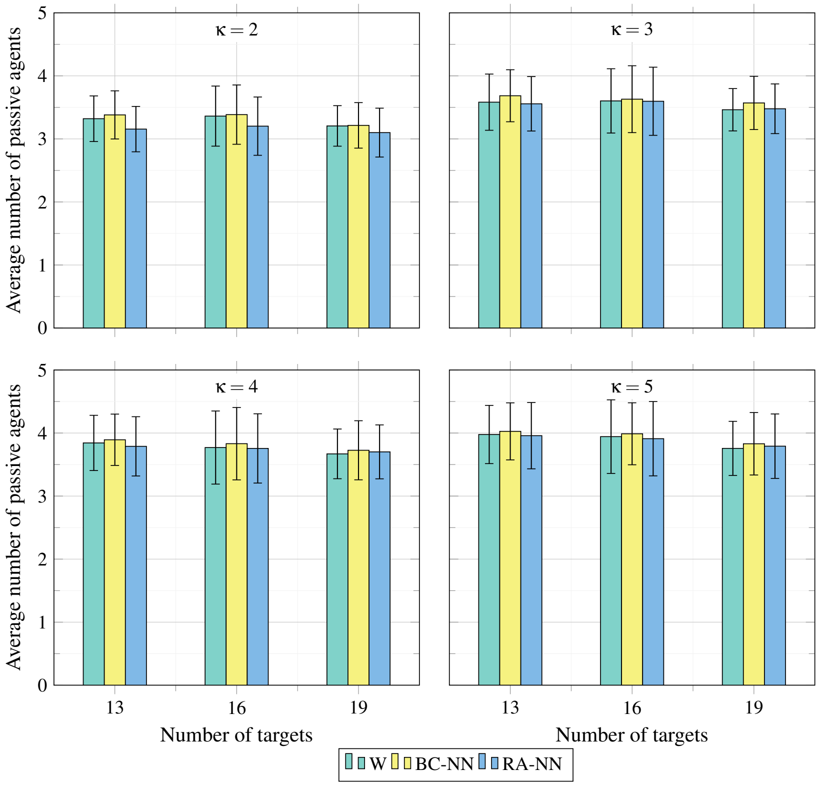

Fig. 10: Average passive number of agents per target for $\kappa \in \{2,3,4,5\}$.

targets, it is assumed that agents know how many targets need to be provisioned at a given a time. This assumption is not made in the other experiments. Depending on how much knowledge agents have access to, the utility functions need to be modified accordingly.

For these experiments, the utility for following a given target is modified to also account for the distance to such target, where the distance is weighed based on the interest level

$$u_{o_i} = \min\left(\frac{1}{(1-I_{o_i}) \cdot d_{a->o_i}}, 1\right), \tag{20}$$

where u_{o_i} represents the utility for following target o_i, I_{o_i} is the interest level of o_i, and $d_{a->o_i}$ is the distance between robot a and target o_i. The intuition behind the equation is that, given the same distance, the utility will be higher for a higher interest level, and lower otherwise.

There are two metrics of interest in these experiments: (i) the average coverage time percentage of all targets $\tau^{(\kappa)}_{\text{avg}\%}$ (Equation 21), and (ii) the average amount of agents covering the targets β_{avg} (Equation 22). The first metric (i) is calculated as

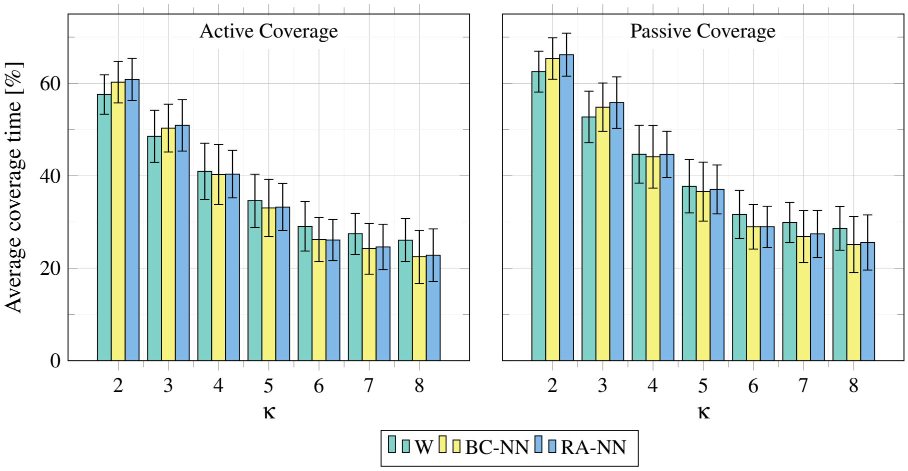

Fig. 11: Average coverage time for active and passive coverage and $\kappa \in \{2,3,4,5,6,7,8\}$.

$$\tau^{(\kappa)}_{\text{avg}\%} = \frac{1}{N}\sum_{e}\frac{1}{|O|}\sum_{j}\tau^{(\kappa)}_{j,e(\%)}, \tag{21}$$

where $\tau^{(\kappa)}_{j,e(\%)}$ is the percentage of time in which the target is covered, calculated as $\frac{c_j^{(\kappa)}}{tp_j}$, where $c^{(\kappa)})_j$ is the amount of time for which the target is covered by κ or more agents, and tp_j is the amount of time for which the target is present in Z (Note that $tp_j <= T_{sim}$). We also compute this metric by weighing $\tau^{(\kappa)}_{j,e(\%)}$ based on the interest level of target o_j, i.e. $\tau^{(\kappa)}_{j,e(\%)} \cdot I_{o_j}$. The second metric (ii) is calculated as

$$\beta_{\text{avg}} = \frac{1}{N}\sum_{e}\frac{1}{|O|}\sum_{j}\beta_{j,e}, \tag{22}$$

where $\beta_{j,e}$ is the average amount of agents that cover target o_j during the part of the simulation in which it is present in Z.

It is possible to observe that the three methods produce comparable results across the different values of κ, considering both active and passive coverage (Figure 11). Slight differences in average are noted, and as κ increases the method with the willingness has a slight improvement compared to the other two. These trends are consistent when weighing $\tau^{(\kappa)}_{j,e(\%)}$ by the interest level of target o_j (Figure 12). Comparing active and passive coverage, it is possible to see that the latter achieves a slight improvement, circa 7% for $\kappa = 2$ (Figure 11).

Regarding the average agents per target, the three methods remain comparable across the different values of κ, considering both active and passive coverage (Figure 13). There are slight differences in average, more notable in the passive case. Fur-

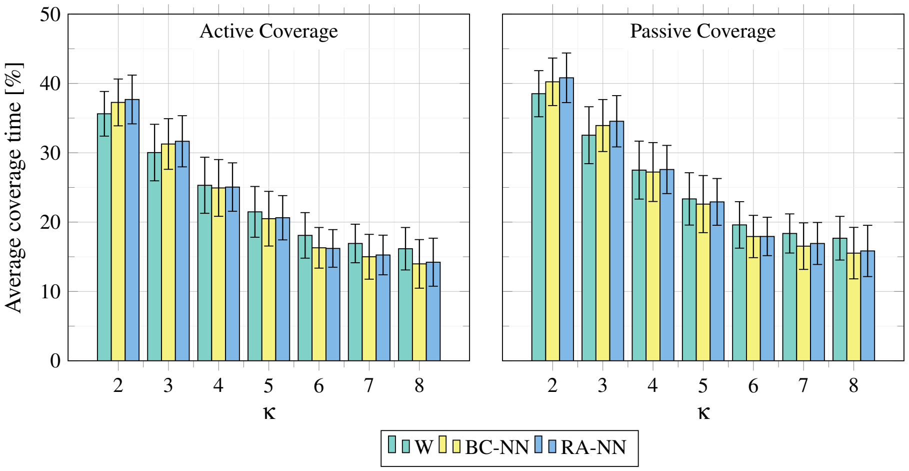

Fig. 12: Average coverage time for active and passive coverage and $\kappa \in \{2,3,4,5,6,7,8\}$, weighed by interest level.

thermore, it is possible to observe that when considering passive coverage as well, the average number of agents per target remains quite close to 3 for all the values of κ under study. This result gives an indication of the physical limit with respect to how many agents can cover a target, given a particular setting, in terms of area size, number of agents, and targets, there is Whereas, in the active case the metric increases with the increase of the value of κ.

6 Generalization of the Approach

In this paper, a collaborative approach based on the willingness to interact has been adopted in order to solve the κ-coverage problem for a multi-robot system. The described framework, composed of the agent behaviour, willingness to interact, combined with the interaction protocols can be applied in other problems as well. Regarding the agent behaviour, the state machine presented in Section 4.2 can be generalised by considering the following abstract states: *idle*, *interact*, *idle & execute*, and *interact & execute* adapted from [11]. Such states can be specialised depending on the desired behaviours robots should display when tackling different problems, e.g. moving by randomly changing direction and inspecting the space for new targets can be used to instantiate the *idle* state into the *inspect* state as done in this paper for solving the κ-coverage problem. Whereas the *execute* state can be instantiated into either the *inspect & follow* or *evaluate & follow*, by adding the target following behaviour to the agents.

The willingness to interact formalism can be easily modified to account for additional relevant factors in a given application domain. Additionally, it is possible to differentiate between different types of factors, such as necessary and optional, as well as giving a specific weight to each factor. In this paper we have considered factors such

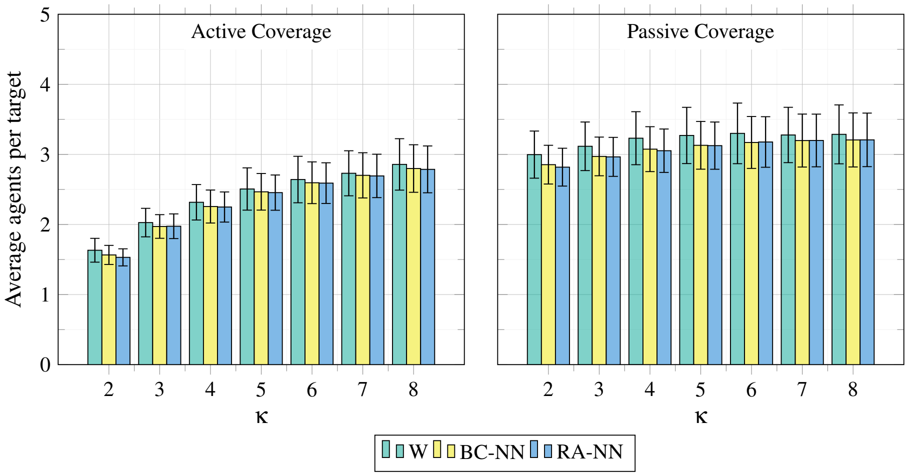

Fig. 13: Average agents per target for active and passive coverage and $\kappa \in \{2,3,4,5,6,7,8\}$.

as the battery level and the level of activity of an agents, which correspond to the necessary and optional factors respectively. Weights are determined in a simple way, i.e. if no necessary factor is under the the minimum threshold, then factors are weighted the same, otherwise the necessary factors will override the optional ones, thus determining the final value of the willingness.

Finally, the interaction protocol is independent of the application and problem to be solved, apart for the κ parameter which can be adjusted depending on the desired size of coalitions for provisioning targets, and the triggers that agents use to initiate the interaction. In the current application domain agents are tasked with discovering and tracking targets in their environment. Therefore, the triggers for executing the interaction protocol are application dependent such as (i) spotting a new target in the visibility range, (ii) detecting that a target is moving away and might soon be outside of the visibility range, and (iii) extending an existing coalition in order to reach κ-coverage. The fourth trigger captures the moment when an agent decides that it needs to ask for help, which is based on the willingness to interact. This trigger is not application dependent.

7 Conclusion and Future Work

This paper presented a novel, distributed, agent-centric coalition formation approach, based on the willingness to interact for adaptive cooperative behaviour. We showed that we can use this novel approach to solve the κ-coverage problem for a group of targets. The performance of this approach is evaluated in two sets of experiments, and is compared with six methods previously proposed in the literature. The purpose of the first set of experiments is to investigate scenarios with static and and mobile targets, while also considering a different number of such targets. For static targets, the analysed metrics are the average time to get one target covered with κ agents, and the average minimum time to κ-cover all objects are considered. Whereas, for mobile targets, we considered the average coverage time and average number of agents per target. Results show that our approach either performs comparably good in the case of static targets with respect to the *BC-NN* method (the best performing among the ones considered in the paper), followed by *RA-NN*, and that it performs better than the other methods in terms of achieving a higher level of coverage when it comes to moving targets. The purpose of the second set of experiments was to further compare the proposed approach with the best performing methods, across several values of κ for mobile targets. Additionally, the performance of the methods was evaluated in scenarios where targets would appear/disappear with a given probability. Results from these experiments show that the three methods are comparable to one another.

There are two main lines of inquiry for future work. First, it is of interest to investigate different rates of appearance and disappearance for targets, in order to understand which method makes for a better coping mechanism in a dynamic environment. Furthermore, issues related to how the studied models scale up in terms of, e.g. bandwidth capacity and latency, can also be considered in the analysis. Second, security aspects can be introduced, by considering the trustworthiness of agents. Such information can be included in the calculation of the willingness to interact, in order to facilitate the cooperation between agents that are more trustworthy, e.g. open systems where new agents may be introduced or removed, similarly to recent approaches [3, 5].

References

1. Bakhshipour, M., Jabbari Ghadi, M., Namdari, F.: Swarm robotics search & rescue: A novel artificial intelligence-inspired optimization approach. Applied Soft Comp. **57**, 708–726 (2017)
2. Burgard, W., Moors, M., Schneider, F.: Collaborative exploration of unknown environments with teams of mobile robots. In: Advances in Plan-Based Control of Robotic Agents. pp. 52–70 (2002)
3. Calvaresi, D., Dubovitskaya, A., Calbimonte, J.P., Taveter, K., Schumacher, M.: Multi-agent systems and blockchain: Results from a systematic literature review. In: Advances in Practical Applications of Agents, Multi-Agent Systems, and Complexity (PAAMS). pp. 110–126 (2018)
4. Casadei, R., Pianini, D., Viroli, M., Natali, A.: Self-organising coordination regions: A pattern for edge computing. In: Coordination Models and Languages. pp. 182–199 (2019)
5. Castelló Ferrer, E.: The blockchain: A new framework for robotic swarm systems. In: Proceedings of the Future Technologies Conference (FTC). pp. 1037–1058 (2019)
6. Eickstedt, D.P., Benjamin, M.R.: Cooperative target tracking in a distributed autonomous sensor network. pp. 1–6. OCEANS 2006 (2006)
7. Elhoseny, M., Tharwat, A., Yuan, X., Hassanien, A.E.: Optimizing k-coverage of mobile wsns. Expert Systems with Applications **92**, 142 – 153 (2018)
8. Esterle, L.: Goal-aware team affiliation in collectives of autonomous robots. In: Int. Conf. on Self-Adaptive and Self-Organizing Systems (SASO). pp. 90–99 (2018)
9. Esterle, L., Lewis, P.R.: Online multi-object k-coverage with mobile smart cameras. In: Int. Conf. on Distributed Smart Cameras. pp. 1–6. ACM (2017)
10. Esterle, L., Lewis, P.R.: Distributed autonomy and trade-offs in online multiobject k-coverage. Computational Intelligence **36**(2), 720–742 (2019)

11. Frasheri, M., Cürüklü, B., Ekström, M., Papadopoulos, A.V.: Adaptive autonomy in a search and rescue scenario. In: Int. Conf. on Self-Adaptive and Self-Organizing Systems. pp. 150–155 (2018)
12. Frasheri, M., Esterle, L., Papadopoulos, A.V.: Modeling the willingness to interact in cooperative multi-robot systems. In: 12th International Conference on Agents and Artificial Intelligence (ICAART). vol. 1, pp. 62–72 (2020)
13. Fusco, G., Gupta, H.: Selection and orientation of directional sensors for coverage maximization. pp. 1–9. Proceedings of the International Conference on Sensor, Mesh and Ad Hoc Communications and Networks (2009)
14. García, S., Menghi, C., Pelliccione, P., Berger, T., Wohlrab, R.: An architecture for decentralized, collaborative, and autonomous robots. In: Int. Conf. on Software Architecture. pp. 75–7509 (2018)
15. Guarnieri, M., Debenest, R., Inoh, T., Fukushima, E., Hirose, S.: Development of helios vii: an arm-equipped tracked vehicle for search and rescue operations. In: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 39–45 (2004)
16. Hefeeda, M., Bagheri, M.: Randomized k-coverage algorithms for dense sensor networks. pp. 2376–2380. Int. Conf. on Comp. Comm. (2007)
17. Hellmund, A., Wirges, S., Ş. Taş, O., Bandera, C., Salscheider, N.O.: Robot operating system: A modular software framework for automated driving. In: Int. Conf. on Int. Transp. Syst. pp. 1564–1570 (2016)
18. Huang, C.F., Tseng, Y.C.: The coverage problem in a wireless sensor network. Mobile Networks and Applications **10**(4), 519–528 (2005)
19. Jung, B., Sukhatme, G.S.: Cooperative multi-robot target tracking. In: Distributed Autonomous Robotic Systems, pp. 81–90 (2006)
20. Khan, A., Rinner, B., Cavallaro, A.: Cooperative robots to observe moving targets: Review. IEEE Trans. Cybernetics **48**(1), 187–198 (2018)
21. King, D.W., Esterle, L., Peterson, G.L.: Entropy-based team self-organization with signal suppression. Artificial Life Conference Proceedings (31), 145–152 (2019)
22. Kolling, A., Carpin, S.: Cooperative observation of multiple moving targets: an algorithm and its formalization. The International Journal of Robotics Research **26**(9), 935–953 (2007)
23. Kumar, S., Lai, T.H., Balogh, J.: On k-coverage in a mostly sleeping sensor network. pp. 144–158. Int. Conf. on Mobile Comp. and Networking (2004)
24. Lee, S.H., Choi, H.: The fast bully algorithm: For electing a coordinator process in distributed systems. In: Int. Conf. on Information Networking, Wireless Communications Technologies and Network Applications (ICOIN). pp. 609–622 (2002)
25. Li, J.S., Kao, H.C.: Distributed k-coverage self-location estimation scheme based on voronoi diagram. IET Communications **4**(2), 167–177 (2010)
26. Liu, B., Dousse, O., Nain, P., Towsley, D.: Dynamic coverage of mobile sensor networks. Trans. Parall. and Distr. Syst. **24**(2), 301–311 (2013)
27. Micheloni, C., Rinner, B., Foresti, G.L.: Video analysis in pan-tilt-zoom camera networks. Signal Processing Magazine **27**(5), 78–90 (2010)
28. Navarro-Serment, L.E., Dolan, J.M., Khosla, P.K.: Optimal sensor placement for cooperative distributed vision. Int. Conf. on Robotics and Automation, vol. 1, pp. 939–944 (2004)
29. Page, J., Armstrong, R., Mukhlish, F.: Simulating search and rescue operations using swarm technology to determine how many searchers are needed to locate missing persons/objects in the shortest time. In: Intersections in Simulation and Gaming: Disruption and Balance. pp. 106–112 (2019)
30. Parker, L.E., Emmons, B.A.: Cooperative multi-robot observation of multiple moving targets. In: Int. Conf. on Robotics and Automation. vol. 3, pp. 2082–2089 (1997)

31. Piciarelli, C., Esterle, L., Khan, A., Rinner, B., Foresti, G.L.: Dynamic reconfiguration in camera networks: A short survey. IEEE Trans. Circuits and Systems for Video Technology **26**(5), 965–977 (2016)
32. Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.Y.: ROS: An open-source robot operating system. In: ICRA workshop on open source software. vol. 3, p. 5 (2009)
33. Qureshi, F., Terzopoulos, D.: Distributed coalition formation in visual sensor networks: A virtual vision approach. In: Int. Conf. on Distr. Comp. in Sensor Syst. pp. 1–20 (2007)
34. Robin, C., Lacroix, S.: Multi-robot target detection and tracking: taxonomy and survey. Autonomous Robots **40**(4), 729–760 (Apr 2016)
35. Ruetten, L., Regis, P.A., Feil-Seifer, D., Sengupta, S.: Area-optimized UAV swarm network for search and rescue operations. In: Computing and Communication Workshop and Conf. (CCWC). pp. 0613–0618 (2020)
36. SanMiguel, J.C., Micheloni, C., Shoop, K., Foresti, G.L., Cavallaro, A.: Self-reconfigurable smart camera networks. Computer **47**(5), 67–73 (2014)
37. Shehory, O., Kraus, S.: Methods for task allocation via agent coalition formation. Artificial Intelligence **101**(1), 165 – 200 (1998)
38. Stormont, D.P.: Autonomous rescue robot swarms for first responders. In: IEEE Int. Conf. Comp. Intell. for Homeland Security and Personal Safety. pp. 151–157 (2005)
39. Theraulaz, G., Bonabeau, E., Deneubourg, J.L.: Response threshold reinforcements and division of labour in insect societies. Royal Society B: Biological Sciences **265**(1393), 327–332 (1998)
40. Werger, B.B., Matarić, M.J.: From insect to internet: Situated control for networked robot teams. Annals of Mathematics and Artificial Intelligence **31**(1), 173–197 (2001)
41. Yanmaz, E., Yahyanejad, S., Rinner, B., Hellwagner, H., Bettstetter, C.: Drone networks: Communications, coordination, and sensing. Ad Hoc Networks **68**, 1–15 (2018)
42. Ye, D., Zhang, M., Sutanto, D.: Self-adaptation-based dynamic coalition formation in a distributed agent network: A mechanism and a brief survey. IEEE Trans. on Parallel & Distributed Systems **24**(5), 1042–1051 (2013)