




Modeling the Willingness to Interact in Cooperative Multi-Robot Systems

Mirgita Frasheri¹^a, Lukas Esterle²^b and Alessandro Vittorio Papadopoulos¹^c

¹Mälardalen University, Västerås, Sweden

²Aarhus University, DIGIT, Aarhus, Denmark

{mirgita.frasheri, alessandro.papadopoulos}@mdh.se, lukas.esterle@eng.au.dk

Keywords: Robot Collaboration, κ -Coverage Problem, Adaptive Autonomy

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 compare it with six other methods from the literature. We investigate the trade-off between the number of robots available and the number of potential targets in the environment. We show that the proposed method is able to provide comparable performance to the best method in the case of static targets, and to achieve a higher level of coverage with respect to the other methods in the case of mobile targets.


1 INTRODUCTION


Robots collaborating with each other in order to tackle a task perform faster and more efficiently than their individually operating counterpart. Some tasks even require collaboration of multiple robots and cannot be accomplished by individual robots at all. However, such collaboration requires coordination of the individual robots, and formation of coalitions between them. Numerous coalition formation approaches have been proposed which either rely on central components [García et al., 2018, Burgard et al., 2002, Shehory and Kraus, 1998] or focus on a single task to be accomplished [Qureshi and Terzopoulos, 2007] in order to achieve meaningful interaction and collaboration. These approaches require dissipation of information about available coalitions as well as negotiations about participation of each potential coalition member [Shehory and Kraus, 1998, Ye et al., 2013, Qureshi and Terzopoulos, 2007]. 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 [Theraulaz et al., 1998, Esterle, 2018]. 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 in order to provision varying sets of tasks efficiently, without a central component involved. We propose an approach where each robot makes individual decisions about whether or not to provision a specific task, employing local information about its own status, e.g., its battery level, its ability, and 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 [Esterle and Lewis, 2017, Esterle and Lewis, 2019], which is related to the cooperative multi-robot observation of multiple moving targets (CMOMMT) problem proposed by Parker and Emmons [Parker and Emmons, 1997], and consists of a varying number of tasks required to be tackled concurrently.

First, the robots need to discover initially unknown moving objects in the environment. They do

^a <https://orcid.org/0000-0001-7852-4582>

^b <https://orcid.org/0000-0002-0248-1552>

^c <https://orcid.org/0000-0002-1364-8127>

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. 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 for the collective of covering this known target. This generates a trade-off between detecting new objects and covering known objects with κ robots when the collective tries to maximize 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 [Robin and Lacroix, 2016], the on-line multi-objective κ -coverage problem is hunting mobile search, monitoring multiple targets, with different viewpoints.

In this paper, we present a novel distributed coalition formation algorithm considering several tasks. At its core, we propose to introduce a *willingness to interact* to each individual robot as the main driver for the coalition formation. The *willingness* is dependent on the state of the robot, such as local conditions like battery level, and current level of activity. Utilizing 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. This approach is evaluated over several scenarios of increasing number of targets, considering both static and mobile targets separately. 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 either comparable with the best of the methods it is compared with – in the case of static objects – while it exhibits a higher coverage in the case of moving targets.

The remainder of this paper is structured as follows. Section 2 gives a formal definition of the on-line multi-object κ -coverage problem and Section 3 covers the behaviour of the agents and targets, their interaction as well as our novel coalition formation algorithm. Section 4 gives an overview of the experi-

mental setup, the performed experiments, and the obtained results. Section 5 discusses the generalization of the proposed approach, while Section 6 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, without any obstacles. We also consider a set of active robots $A = \{a_1, a_2, \dots, a_n\}$, and a set of targets or objects of interest $O = \{o_1, o_2, \dots, o_m\}$ in this problem. Both robots and objects can freely move within Z , with (nonconstant, yet limited) velocities v_i , where $i = 1, \dots, n$, and v_j , where $j = 1, \dots, m$; 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, i.e., targets cannot appear or disappear. 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 can be perceived 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 associated with a prescribed constant interest level l_j . Levels of interest are not necessarily the same between objects, and define 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 (as detailed in Section 3.3) at each time-step. This can occur in different situations, e.g., (i) when a robot i first detects an object j entering or leaving its sensing area an object j is entering the sensing area of the robot i and (ii) when robot i receives an invitation to provision an object j from another robot. Robots are assumed to communicate with one another via broadcast, as implemented in ROS [Quigley et al., 2009]. Thus, the willingness to interact shapes the cooperative behavior of an agent and its respective robot in relation to the others. Robots are able to change and keep track of their own state and behavior, as well as the state and behavior 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 robot-independent, i.e., there is a shared

knowledge among the agents on the level of interest of different targets.

The online multi-object κ -assignment problem is solved by having at least κ robots covering any target in the set. Consequently, two tasks should be achieved concurrently: (i) maximizing the number of provisioned objects, and (ii) provisioning 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 cover all targets moving around 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. Is the motion of the targets affecting the obtained performance?
4. How does the defined value for κ affect the performance of the robot cooperation?
5. How does the proposed method compare with other state-of-the-art techniques for the κ -coverage problem?

We address these questions using two experimental setups with varying number of either mobile or static (immobile) targets, according to metrics that analyze 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 [Esterle and Lewis, 2017].

3 AGENT MODEL

In this section, we describe how a robot operates, how the agent, embodied in a robot, updates the willingness to interact, and how these agents form decisions to cooperate through the proposed interaction protocols. In the following we are using the terms robot and agent interchangeably.

3.1 Robot Kinematics

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

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

where $x_a(t)$ and $y_a(t)$ are the x- and y-coordinate on the map and define the position $\mathbf{p}_a = (x_a, y_a)$ 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 localize 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 \mathbf{d}_a over which the robot moves is thus computed as

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

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

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

$$\tilde{v}_a(t) = \|\mathbf{d}_a(t)\| \quad (4)$$

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

$$e_a(t) = \theta_a^\circ(t) - \theta_a(t) \quad (5)$$

$$\tilde{\omega}_a(t) = K_p \text{atan2}(\sin(e_a(t)), \cos(e_a(t))) \quad (6)$$

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

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

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

3.2 Agent Behavior

Software agents, embodied in physical robots, operate autonomously and their behavior can be described as a state machine composed of four states: *inspect*, *evaluate*, *inspect & follow*, and *evaluate & follow*. Figure 1 shows the state machine that describes the behavior structure of an agent. At run-time, any agent starts its operation in the state *inspect*, in which it moves 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 the *evaluate* state, an agent decides how it wants to interact with the spotted target or the request for help, based on its current state. The proposed interaction protocol is described in detail in Section 3.4. The result of the interaction is a coalition of agents that will start following the spotted target. If the agent is not part of the coalition after the interaction, it will switch back

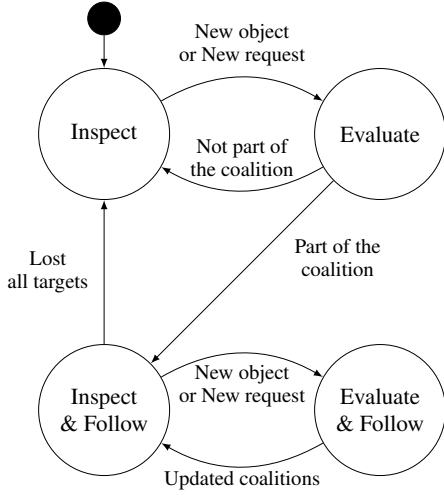


Figure 1: Agent operation state machine.

to the *inspect* state, looking for other targets in the environment. 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. In case an agent loses track of all the targets it is following, then it switches to *inspect*. In case an agent 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 this state, the agent either generates a help request, or it responds to a help request. In both cases, the agent decides if it will be part of a new coalition, or if it is going to drop a target. 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 simultaneously 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 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.

3.3 Willingness to Interact

The willingness to interact w shapes the cooperative behavior of an agent, i.e., when an agent should ask for help and when it should give help. This parameter does not refer to a particular task that should be completed, but rather reflects the general disposition of any agent to cooperate with the others. The willingness to interact w takes values in $[-1, 1]$. When $w \geq 0$ the agent is willing to provide support to other agents that have requested help. When $w < 0$, the agent will raise 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:

$$w_i(t+1) = \min(\max(w_i(t) + \mathbf{B}^\top \mathbf{f}(t), -1), 0), \quad (9)$$

$\mathbf{f}(\cdot) = [f_1(\cdot), \dots, f_m(\cdot)]^\top$ is an $m \times 1$ vector of the m factors that affect the willingness, while $\mathbf{B} = [\beta_1, \dots, \beta_m]^\top$ 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

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

where $\phi(k)$ represents the current measurement of that factor (e.g., the current battery level), while ϕ_{\min} is a minimal threshold considered acceptable (e.g., the minimal battery level to perform a task). The terms ϕ and ϕ_{\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 reach the moving target, or to complete an assigned task. Therefore, an agent with a low battery level should try to receive help from the other agents. On the other hand, the number of targets (n_O) an agent is tracking is considered as optional, since an agent can follow several targets, but this makes its task 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 is defined by their corresponding weights. The

weight for a necessary factor β_{nec} is defined as:

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

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

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

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. In such a case the calculation for factors and weights has to be adapted accordingly. More examples on factors that can affect the willingness can be found in [Frasheri et al., 2018].

3.4 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 SCR design pattern [Casadei et al., 2019], 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 in case it spots a new target, or it wants to extend an existing coalition to reach κ -coverage, or it perceives that targets in its visibility range are moving away from itself, and if it is necessary to ask for help (e.g., battery level is under the accepted minimum). Furthermore, agents can decide 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, it broadcasts an *information request* to other agents together with its willingness and respective utility for provisioning the target. 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 exists already for the given target, the corresponding leader will reply whether or not further agents are needed to reach the κ -coverage. If no help is needed, then the agent continues its previous activities. If help is needed, then the agent will receive an assignment

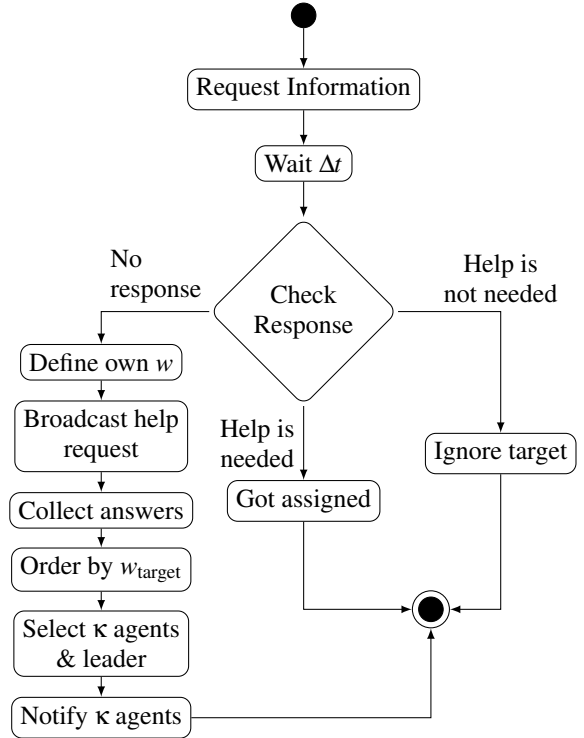


Figure 2: Activity diagram of the agent's behavior when a new target is spotted.

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 no other agent is following 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 from following the target. The mechanism follows the logic of a fast bully algorithm [Lee and Choi, 2002], well known in distributed systems. A request for help to follow the object is broadcast to all other agents. Other agents send 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 not considered further. 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 :

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

Utilities are defined by each agent for the individual target and can generally vary between the different agents as well as the different targets. Examples for this could be the size, speed, or direction of movement of the object. In our experiments, we consider dif-

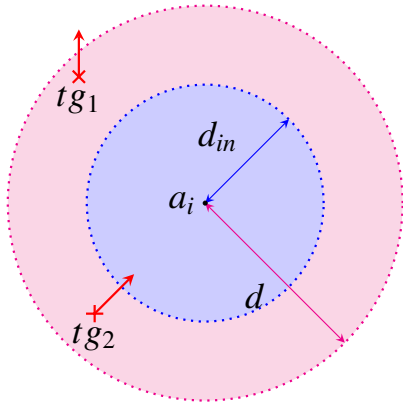


Figure 3: Agent a_i with visibility range d , indicated with the red circle, and internal range d_{in} , indicated with the blue circle. The targets tg_1 and tg_2 are indicated with crosses, and they are moving towards and away from the agent, respectively.

ferent 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, 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. The initiating agent does not necessarily need to be part of the coalition.

Furthermore, every agent keeps track of whether the targets in its visibility range are moving away from the robot. We introduce another internal threshold with radius d_{in} around the robot, where $d_{in} < d$ (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. At the same time, an agent will consider 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. If a leader agent needs to replace itself, then 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, if a common agent needs to drop a target, then it first notifies its leader. Leaders are also responsible for triggering continuously the extension of a coalition in order to

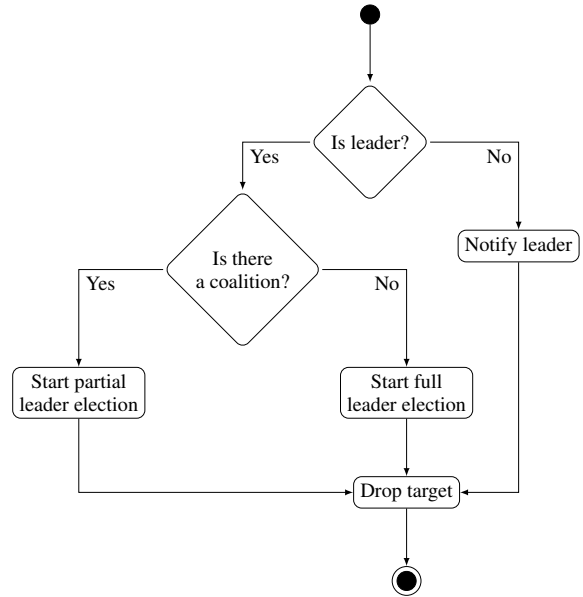


Figure 4: Activity diagram of the agent's behavior when it needs to replace itself in a coalition.

maintain κ -coverage, following a monotonically increasing period.

4 SIMULATION SETUP

The behavior of the agents was evaluated with computer simulations¹, based on the robot operating system (ROS) [Quigley et al., 2009, Hellmund et al., 2016] to model the agents kinematics, behavior, and interaction.

The method utilizing the willingness to interact, as described in this paper, was compared to six other methods that were previously proposed in the literature for solving the multi-object κ -coverage problem [Esterle and Lewis, 2017]. Each of the six methods is a combination of one communication model and one response model. Two communication models are considered, broadcast *BC* and random *RA*. In the broadcast model an agent broadcasts help request to everyone, whereas in the random model it sends a help request to κ random agents. As for the response models, three are considered: (i) newest-nearest *NN*, (ii) available *AV*, and (iii) received calls *RE*. In the newest-nearest model an agent will answer to the request that is newest, and if there are multiple request at the same time, it will respond to the one that is nearest. In the available model the agent answers to

¹The code for running the simulations is publicly available at https://gitagent@bitbucket.org/gitagent/gitagent_2.git

requests according to the newest-nearest strategy only if it is not engaged in following other target(s). In received calls, 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 are: *BC-NN*, *BC-AV*, *BC-RE*, *RA-NN*, *RA-AV*, and *RA-RE*. Such selection is due to previous results [Esterle and Lewis, 2017], where the broadcast and random communication models were evaluated better with respect to the rest, and the response models were reported to have a significant impact on the κ -coverage.

In all of our 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., we are considering a limited area surrounded by walls. The objects to be covered are distributed uniformly in the map Z of size $100\text{m} \times 100\text{m}$. We consider 7 different scenarios for our experiments 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 analyzed the behavior of our agents to achieve κ -coverage where $\kappa \geq 3$, and $\kappa \geq 5$.

4.1 Results for Static Targets

In the first set of experiments, we consider only targets that remain in their initial location ($v_j = 0$). Once the 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 $t_{j,e}^{(1)}$, and the time to reach κ -coverage $t_{j,e}^{(\kappa)}$. Based on this information we can calculate: (i) the average time to get one target to be covered by at least by κ agents $t_{\text{avg}}^{(\kappa)}$, and (ii) the average minimum time to get all the targets covered by at least κ agents $t_{\text{min}}^{(\kappa)}$. These two metrics give an indication of a minimum coverage, and a complete coverage, and the respective timing

properties. They are formally defined as:

$$t_{\text{avg}}^{(\kappa)} = \frac{1}{N} \sum_e \frac{1}{|O|} \sum_j t_{j,e}^{(\kappa)} \quad (14)$$

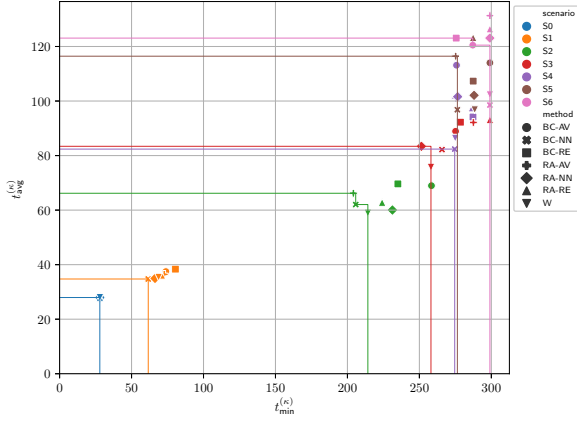
$$t_{\text{min}}^{(\kappa)} = \frac{1}{N} \sum_e \max_j t_{j,e}^{(\kappa)} \quad (15)$$

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

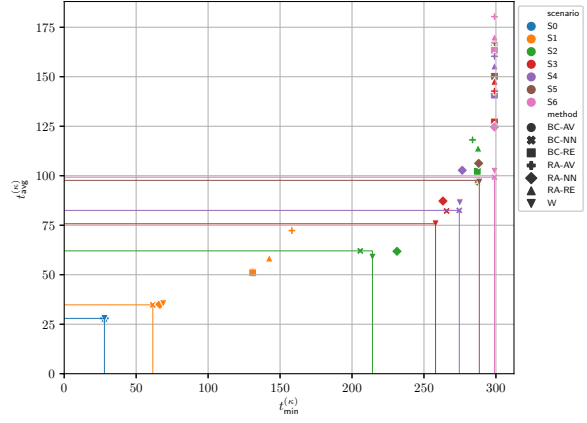
Results for $\kappa \geq 3$ as well as $\kappa \geq 5$ are shown in Figure 5, where $t_{\text{min}}^{(\kappa)}$ is given on the x -axis, and $t_{\text{avg}}^{(\kappa)}$ is given on the y -axis. In both metrics, the lowest value, the better. In the graph we also indicate the corresponding Pareto frontier 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 utilized methods for scenario $S0$ for $\kappa \geq 1$, as shown in Figure 5. This is due to the fact that there is only one static target in the environment, which will be discovered at the exact same time irrespective of the 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 in the ROS service calls or broadcast used by the agent 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 $S1$ – $S6$, there is a difference on average between the different methods. 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 5b; $S2$ and $S5$ in Figure 5c; $S2$, $S5$, and $S6$ in Figure 5d; and for scenarios $S2$ and $S3$ in Figure 5a; $S2$ in Figure 5b; $S4$ and $S6$ in Figure 5c; 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.

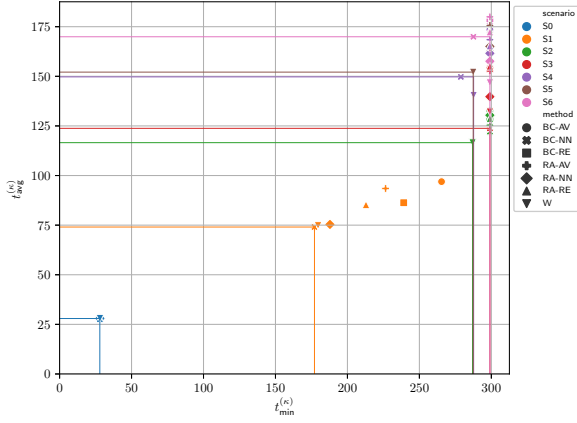
When only one target is involved, the average minimum time to coverage is lowest. In all cases, the



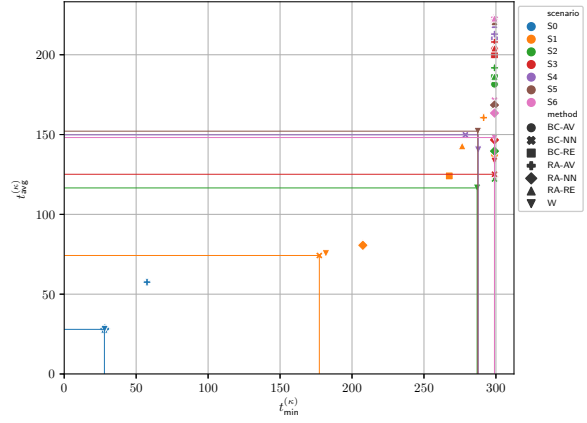
(a) Average vs Minimum Time to Coverage with at least one agent when tasked with $\kappa \geq 3$.



(b) Average vs Minimum Time to achieve $\kappa \geq 3$.



(c) Average vs Minimum Time to Coverage with at least one agent when tasked with $\kappa \geq 5$.



(d) Average vs Minimum Time to achieve $\kappa \geq 5$.

Figure 5: Average time vs minimum time to cover all stationary targets (i.e., not moving) with 1 (left) or κ agents (right). The top row show results where agents are tasked to cover targets with $\kappa \geq 3$ while the lower row shows results where agents are tasked to cover targets with $\kappa \geq 5$.

agents will move in Z and eventually find and cover the targets. However, when increasing the number of targets (S1–S6), it can happen that agents gather on the first targets found, leaving remaining targets undiscovered for the rest of the simulation. As such, all metrics are affected, and the $t_{j,e}^{(\kappa)}$ is saturated to the duration of the simulation $T_{\text{sim}} - 1$ ² for the targets that were not discovered. In Figure 5, the points are accumulated at the $t_{\text{min}}^{(\kappa)} - 1$, which means that in those scenarios there were undiscovered targets for the whole duration of the simulation.

²In the final time-step the multi-agent system shuts down, hence this time-step is not considered when dealing with the results.

4.2 Results for Dynamic Targets

In our second set of experiments, targets move within the map Z by randomly changing direction, with velocity $v_{l,\text{max}} = 1.5 m$ 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 average time for which a target j is covered with at least κ agents over the simulation, $\tau_{j,e}^{(\kappa)}$, 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, $\tau_{\text{avg}}^{(\kappa)}$, and (ii) the average amount of agents that cover the targets α_{avg} . These

quantities are computed as

$$\tau_{\text{avg}}^{(\kappa)} = \frac{1}{N} \sum_e \frac{1}{|O|} \sum_j \tau_{j,e}^{(\kappa)} \quad (16)$$

$$\alpha_{\text{avg}} = \frac{1}{N} \sum_e \frac{1}{|O|} \sum_j \alpha_{j,e} \quad (17)$$

While in our first set of experiments, featuring a set of static targets, agents will cover a target for the whole duration of the simulation once they joined a coalition, 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 that agents are actively following by adjusting their own motion and being part of a coalition, and (ii) considering targets that are not being actively followed, but are within the visibility range of agents, without necessarily being part of a coalition.

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. It is possible to observe that for $\kappa \geq 3$ the method with the willingness is overall on the Pareto frontier, with an exception for scenario S3, shown in Figure 6a. Regarding $\kappa \geq 5$, the method with the willingness, indicated with W in the legends of Figure 6, is on the Pareto frontier for scenarios S0–S4, and the best on average for S5–S6, Figure 6c. The same is observed for passive following in Figure 6d. Furthermore, our approach tends toward maximizing 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, whilst the size of the area is unchanged, might increase the average coverage time as agents can join multiple coalitions. 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. As such the lack of coverage before the discovery naturally penalizes the shown results.

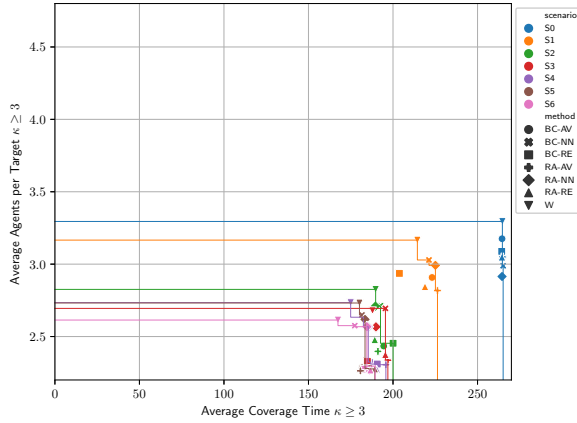
In our current approach, the agents are not aiming to exceed the desired coverage. Nevertheless, this can happen due to race-conditions in the coalition formation process. Furthermore, this can also take place when an agent detects that a target is moving away. In this case it will try to find another agent that can join the coalition. At the same time, the target is not dropped by the former agent until it actually goes out of its visibility range while the new agent already joined the coalition and might have the target within its visible range.

5 GENERALIZATION OF THE APPROACH

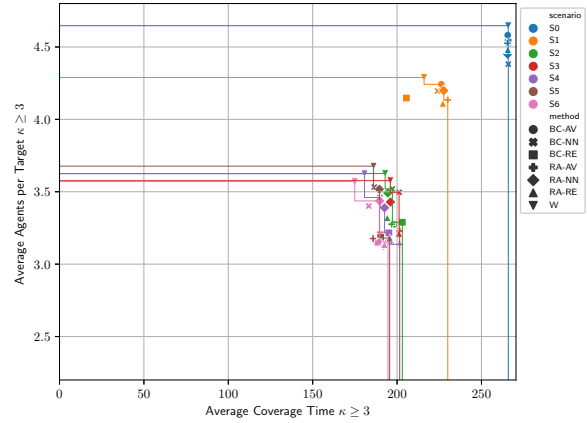
In this paper, a collaborative approach based on the willingness to interact has been tailored to solve the κ -coverage problem for a multi-robot system. The described framework, composed of the agent behaviour, willingness to interact, and interaction protocol can be applied in other problems as well. Regarding the agent behaviour, the state machine presented in Section 3.2 can be generalized by considering the following abstract states: *idle*, *interact*, *idle & execute*, and *interact & execute* adapted from [Frasheri et al., 2018]. The latter can be specialized depending on the behaviours that the robots should have for solving 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 adopted to account for additional relevant factors in a given application domain. The framework allows for the factors to be grouped into two categories, necessary and optional, as well as giving a specific weight to each factor. In this paper we have considered the battery level and the number of targets an agent is already following, 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 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.

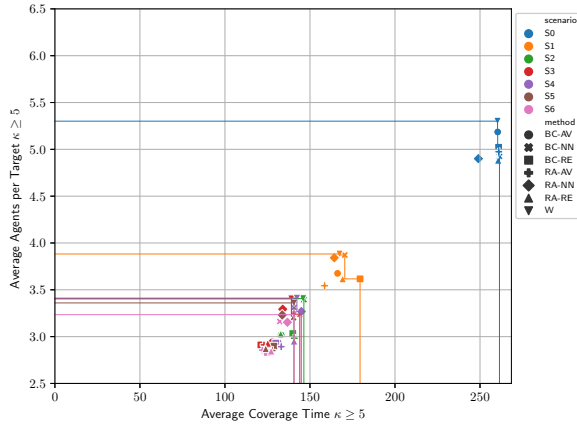
Finally, the interaction protocol is independent of the application and problem to be solved, apart for the κ parameter which can be adjusted depending on the size of the coalitions that the robots should be able to



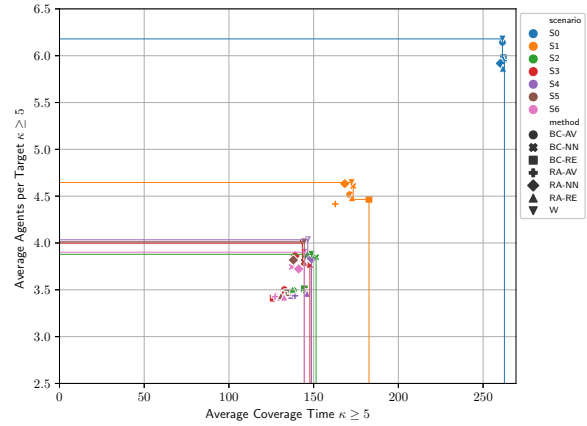
(a) Active Coverage $\kappa \geq 3$



(b) Passive Coverage $\kappa \geq 3$



(c) Active Coverage $\kappa \geq 5$



(d) Passive Coverage $\kappa \geq 5$

Figure 6: Average number of agents covering targets vs average coverage time of all targets of the entire duration of the simulation. On the left we show active coverage where we only consider the agents actively following a target (left) while the right includes passive coverage - agents having multiple targets in their visibility range while following another target. The top row show results where agents are tasked to cover targets with $\kappa \geq 3$ while the lower row shows results where agents are tasked to cover targets with $\kappa \geq 5$.

form, 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.

6 CONCLUSION AND FUTURE WORK

This paper presented a novel, distributed, agent-centric coalition formation approach, based on the willingness to interact for adaptive cooperative behavior. We showed that we can use this novel approach to solve the κ -coverage problem for a set of targets. The performance of this approach is measured along two different sets of metrics for two different cases, (i) with static targets, and (ii) with mobile targets, and compared with six methods previously proposed in the literature. In the former case, the average time to get one target covered with κ agents, and the average minimum time to κ -cover all objects are consid-

ered. In the latter case, the average coverage time and average number of agents per target are considered. 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), and that it performs better than the other methods in terms of achieving a higher level of coverage when it comes to moving targets.

There are three main lines of inquiry for future work. First, it is of interest to compare further the performance of our approach with those methods that reached similar performance like the *BC-NN* method. Such investigation can include the exploration of other experimental settings that might better highlight possible trade-offs for the utilization of the *BC-NN* method or the one based on the willingness proposed in this paper. 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 [Castelló Ferrer, 2019, Calvaresi et al., 2018]. Third, some assumptions made in this paper can be relaxed, e.g., targets can appear and disappear at random times, or leave the area defined by the map, in order to adapt the current approach for solving a more general k -coverage problem.

ACKNOWLEDGEMENTS

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.

REFERENCES

- Burgard, W., Moors, M., and Schneider, F. (2002). Collaborative exploration of unknown environments with teams of mobile robots. In Beetz, M., Hertzberg, J., Ghallab, M., and Pollack, M. E., editors, *Advances in Plan-Based Control of Robotic Agents*, pages 52–70.
- Calvaresi, D., Dubovitskaya, A., Calbimonte, J. P., Taveter, K., and Schumacher, M. (2018). Multi-agent systems and blockchain: Results from a systematic literature review. In Demazeau, Y., An, B., Bajo, J., and Fernández-Caballero, A., editors, *Advances in Practical Applications of Agents, Multi-Agent Systems, and Complexity: The PAAMS Collection*, pages 110–126, Cham. Springer International Publishing.
- Casadei, R., Pianini, D., Viroli, M., and Natali, A. (2019). Self-organising coordination regions: A pattern for edge computing. In Riis Nielson, H. and Tuosto, E., editors, *Coordination Models and Languages*, pages 182–199, Cham. Springer International Publishing.
- Castelló Ferrer, E. (2019). The blockchain: A new framework for robotic swarm systems. In Arai, K., Bhatta, R., and Kapoor, S., editors, *Proceedings of the Future Technologies Conference (FTC) 2018*, pages 1037–1058, Cham. Springer International Publishing.
- Esterle, L. (2018). Goal-aware team affiliation in collectives of autonomous robots. In *Proc. of the Int. Conf. on Self-Adaptive and Self-Organizing Systems (SASO)*, pages 90–99.
- Esterle, L. and Lewis, P. R. (2017). Online multi-object k-coverage with mobile smart cameras. In *Proc. of the Int. Conf. on Distributed Smart Cameras*, pages 1–6. ACM.
- Esterle, L. and Lewis, P. R. (2019). Distributed autonomy and trade-offs in online multiobject k-coverage. *Computational Intelligence*.
- Frasheri, M., Cürüklü, B., Ekström, M., and Papadopoulos, A. V. (2018). Adaptive autonomy in a search and rescue scenario. In *Proc. of the Int. Conf. on Self-Adaptive and Self-Organizing Systems*, pages 150–155.
- García, S., Menghi, C., Pelliccione, P., Berger, T., and Wohrab, R. (2018). An architecture for decentralized, collaborative, and autonomous robots. In *Proc. of the Int. Conf. on Software Architecture*, pages 75–7509.
- Hellmund, A., Wirges, S., Ş. Taş, O., Bandera, C., and Salscheider, N. O. (2016). Robot operating system: A modular software framework for automated driving. In *Proc. of the Int. Conf. on Intelligent Transportation Systems*, pages 1564–1570.
- Lee, S.-H. and Choi, H. (2002). The fast bully algorithm: For electing a coordinator process in distributed systems. In *Revised Papers from the International Conference on Information Networking, Wireless Communications Technologies and Network Applications-Part II, ICOIN '02*, pages 609–622, Berlin, Heidelberg. Springer-Verlag.
- Parker, L. E. and Emmons, B. A. (1997). Cooperative multi-robot observation of multiple moving targets. In *Proc. of the Int. Conf. on Robotics and Automation*, volume 3, pages 2082–2089 vol.3.
- Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R., and Ng, A. Y. (2009). ROS: An open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5.
- Qureshi, F. and Terzopoulos, D. (2007). Distributed coalition formation in visual sensor networks: A virtual vision approach. In Aspnes, J., Scheideler, C., Arora, A., and Madden, S., editors, *Proc. of the Int. Conf. on Distributed Computing in Sensor Systems*, pages 1–20.

- Robin, C. and Lacroix, S. (2016). Multi-robot target detection and tracking: taxonomy and survey. *Autonomous Robots*, 40(4):729–760.
- Shehory, O. and Kraus, S. (1998). Methods for task allocation via agent coalition formation. *Artificial Intelligence*, 101(1):165 – 200.
- Theraulaz, G., Bonabeau, E., and Deneubourg, J.-L. (1998). Response threshold reinforcements and division of labour in insect societies. *Proc. of the Royal Society B: Biological Sciences*, 265(1393):327–332.
- Ye, D., Zhang, M., and Sutanto, D. (2013). 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.