A distributed approach for autonomous cooperative transportation in a dynamic multi-robot environment

Amar Nath[1]
anath@cs.iitr.ac.in
IIT Roorkee - 247667
India

Arun AR[2]
arunnadar1994@gmail.com
IIT Roorkee - 247667
India

Rajdeep Niyogi[3]
rajdpfec@iitr.ac.in
IIT Roorkee - 247667
India

ABSTRACT

This paper describes a distributed approach for autonomous cooperative transportation in a dynamic multi-robot environment. The proposed approach forms an optimal coalition at runtime for cooperative transportation and assigns a group of robots for the task. Explicit communication is used to acquire information as the robots do not have a global knowledge of the environment, i.e., no robot knows the location and state of another robot. For cooperative transportation, such information is essential as objects may arrive at any time and at any location. The proposed approach deals with on-demand missions, where the number of robots required to solve the problem is not known a priori. The applicability of the approach is demonstrated on a road clearance scenario in a realistic urban search and rescue simulation environment. The experimental results validate the correctness of the approach.

CCS CONCEPTS

• **Multi-agent systems** → **Multi-robot systems**; • **Multi-robot coordination**; • **Coalition/team formation** → *Cooperative goal achievement*; • **Distributed algorithm**;

KEYWORDS

Multi-robot system, coperative object transportation, coalition formation, distributed algorithm, multi-robot coordination

ACM Reference Format:
Amar Nath[1], Arun AR[2], and Rajdeep Niyogi[3]. 2020. A distributed approach for autonomous cooperative transportation in a dynamic multi-robot environment. In *SAC IRMS '20: ACM/SIGAPP Symposium on Applied Computing, March 30–April 03, 2020; Brno, Czech Republic.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3341105.3373980

1 INTRODUCTION

Autonomous mobile robots have now emerged as a means of transportation in several applications, such as, warehouse, factory, space, and deep-sea where direct human intervention is impossible or impractical [1, 17]. Although multi-agent system research has received substantial attention, multi-robot coordination remains a challenging problem since the overall performance of the system is directly affected by the quality of coordination and control among the robots while executing cooperative tasks [4]. Coordination in a multi-robot system can be achieved either by explicit or by implicit communication [17, 21]. Since explicit communication provides a better and reliable way of multi-robot coordination compared to implicit communication [17, 21], so it is preferred in critical missions, such as, search and rescue, where efficient and continuous coordination between robots is required.

Cooperative object transportation is needed when the object is either heavy or too large or needs extra care to handle (e.g., shifting a glass table) or has a complex shape which makes it difficult for a single robot to transport. All group members need not participate in the physical act of transport; cooperation can still be achieved when some robots transport the object and others are involved in, say, coordination and navigation along the desired trajectory and/or clear obstacles along the path [9].

The problem of cooperative object transportation involves several challenges, e.g., the location and time of object arrival, the number of robots required to transport the object, states, and location of the robots are not known in advance in a dynamic environment. The problem of cooperative transportation discussed in this paper falls in the category of ST-MR-IA (ST means a robot can execute at most a single task at a time, MR means that some tasks can require multiple robots, IA stands for instantaneous allocation (i.e., at runtime) of robots for the tasks) as per the taxonomy given in [6]. Cooperative object transportation requires different types of skills, such as pushing, lifting, vision, and grabbing, and so a heterogeneous rather than a homogeneous group of robots is required [18]. Homogeneous groups are frequently found in swarm robotics, where the robots mimic the behavior of insects, such as ants and bees [16].

To the best of our knowledge, most of the existing literature considers the problem of cooperative object transportation with the following assumptions, namely, the source and destination of the object and the number of robots required to transport the object are known in advance. However, we are interested in solving the cooperative object transportation problem in a dynamic environment automatically without any intervention of a central agent or a human, where the source and target location, number of robots required for transportation, roles of the robots, time of task arrival are not known in advance. We suggest a distributed approach for solving the cooperative object transportation problem in a dynamic environment, that uses explicit communication to coordinate the activities of the robots. The distributed algorithm forms a coalition at runtime and assigns the task to members of the coalition. Then the coalition performs cooperative object transportation. We demonstrate the applicability of the proposed approach with a

SAC IRMS '20, March 30–April 03, 2020, Brno, Czech Republic

https://doi.org/10.1145/3341105.3373980

road clearance scenario in an urban search and rescue simulation environment using ARGoS, a realistic multi-robot simulator [15].

The remaining part of the paper is organized as follows. Related work is given in Section 2. The problem formulation is given in Section 3. The proposed algorithm is described in Section 4. In Section 5, the experimental results are provided and analyzed. The conclusions are given in Section 6.

2 RELATED WORK

In the literature, several approaches have been suggested for solving the problem of cooperative object transportation [2, 3, 6, 11, 14, 17, 19]. The work [11] is considered as the pioneering work, targeting a cooperative transport task by a homogeneous group of simple robots that can only push an object. The authors [11, 17] demonstrate that coordinated effort is not possible without explicit communication.

The work [14] proposed direct (explicit) communication to improve the coordination of a homogeneous group of two six-legged robots required to transport a rectangular box toward a target cooperatively. The work [5] considered the problem of cooperative box pushing where the roles of the members are pre-defined; specifically one robot acts as a watcher and the others act as pusher. However, we consider a more complex scenario of cooperative object transportation scenario, where the role of each robot is not fixed in advance, rather decided at runtime. In [3], the robots are designed to push the object across the portion of its surface, where it occludes the direct line of sight to the goal. This simple behavior results in transporting the object towards the goal without using any form of direct communication.

The work [19] suggest a centralized approach, where the roles of the agents are fixed, to cooperatively transport a box to the destination by removing obstacles on its way. Three different types of agents for vision, learning, and physical are used in [19] that performs their corresponding duties; it does not explore the process of team formation and gathering of the robots to the object's location. A recent work [2] demonstrates an effective coordination of actions for initiating and sustaining the transport of heavy objects. The algorithm [2] runs on a server that controls the activities of the robots required to push an object on circular and straight trajectories. In our work the robots are autonomous and control their activities by interacting with each other via exchanging messages. The destination location is also decided by a robot who has detected a task.

The problem of cooperative transportation, considered in this paper, involves team formation of heterogeneous robots and gathering the robots to the location of the object to be transported. The number of robots required to transport the object is not known a priori and it is decided at runtime. For the same task, the team size is determined by the state of the environment. So, at some point in time an object may be transported by two robots while at some other moment in time three or more robots are required. Few works related to coalition formation strategies are discussed below.

Existing approaches such as [8, 12] reduce the team formation problem to a combinatorial optimization problem, where the objective is to select k agents from a set of N agents, where each agent has some skills, such that a given cost criterion is optimized. These approaches [8, 12] assume complete knowledge of the environment (e.g., the total number of agents in the environment, the skill of an agent); current state and location of an agent are irrelevant in these approaches. Typically, in multi-agent settings, agents have limited information about an environment (e.g., the total number of agents in the environment, the skill of an agent), and they should coordinate among themselves (say, by exchanging messages) to accomplish a given task.

Auction-based approaches for team formation (task allocation) are suggested in [5, 10, 20]. A bidder agent has some resources (e.g., data center, CPU) [10], who may bid for multiple auctioneers concurrently. However, when we move to physical agents, a robot cannot be a member of multiple coalitions at any point of time simply because the tasks may be at different locations, and a robot cannot be at two different locations at the same time, even though a robot may have the capability to perform multiple tasks at a time.

In our work, a non-initiator robot (bidder) will not express its willingness to multiple initiators (auctioneers) concurrently; when more than one request message arrives, the robot stores the requests in its local queue. Having one or more resources specified in the auction is a sufficient condition for an agent to make a bid [10]. Having the required skills for a task is a necessary but not a sufficient condition for a robot to express its willingness to be part of a team, in our work. A robot's behavior, in our work, is determined by its current state, whereas in [5, 10] states need not be taken into consideration.

In [7], the authors describe a framework for dynamic heterogeneous team formation for robotic urban search and rescue. The task discovery is made by a member of a team and it is sent to the team coordinator for assignment. The team coordinator performs the task assignment ensuring the task will be carried out by a robot with the necessary capabilities. However, in a distributed system, no robot knows the states, locations, and skills of other robots. Thus, the robots should communicate among themselves to acquire relevant information for task execution without the intervention of any central authority. This necessitates the design of a distributed algorithm for task execution in such a dynamic environment.

In our approach, unlike [7], every robot has a similar level of priority, and each of them can perform the task management activities, i.e., searching, team/coalition formation by acquiring the information from the robots available in the environment at that moment in time. In this paper, the arrival time and location of a task are not known a priori; hence, task searching and coalition formation activities are performed by a robot at runtime. The details of the proposed algorithm are given in Section 4.

3 PROBLEM FORMULATION

In this section we give a formal framework of a dynamic environment and some related concepts.

Definition 3.1. (**Dynamic Environment**) A global view (snapshot) of an environment E, with a set of locations L, taken at time t, is given by a 3-tuple $E^t = \langle R^t, M^t, loc \rangle$ where R^t is the set of robots present in the environment at time t, and M^t is a mission (set of tasks) in the environment at time t, $loc : R^t \times \mathbb{N} \rightarrow L$, is a function that gives the location of a robot at a discrete instant of time represented by the set of timestamps $\mathbb{N}$.

The robots are assumed to be autonomous and collaborative in the sense that they express their willingness to be part of a team. The states of a robot are IDLE, ANALYZE, PROMISE, BUSY. The significance of the states are: IDLE means that a robot is not executing any task and it is free to take a task; BUSY means that a robot is engaged in executing a task; ANALYZE means that a robot has detected a task and it has started the coalition formation process; and PROMISE means that a robot has expressed its willingness to be part of a coalition. In a dynamic environment, the state of a physical agent (robot) is crucial as a robot can participate in one task (e.g., cooperative object transpiration) and can be present at one location, at a time. A robot can enter the environment at any point in time, but can leave only if it is in IDLE state.

Each robot r has a p-dimensional binary skill vector, $\psi_r = [\alpha_1, \ldots, \alpha_p]$. The value of a particular skill $\alpha_i \in \psi_r$, $i = 1, \ldots, p$, may be 1 or 0 depending on whether r possess a particular skill or not. We assume that for the execution of any task, one or more skills may be required. An example of representing skills using a skill vector is as follows. Let $p = 3$; the first element of the vector denotes griping, the second denotes vision, and the third denotes pushing. Let r_1 and r_2 possess one skill each, e.g., pushing, and vision respectively. The skill vector for robot r_1 and r_2 can be represented as: $\psi_{r_1} = [0\ 0\ 1]$, and $\psi_{r_2} = [0\ 1\ 0]$. Let r_3 and r_4 have skills $\psi_{r_3} = [1\ 1\ 1]$ and $\psi_{r_4} = [1\ 0\ 0]$ respectively. The skill vector for the robots may be represented as shown in Table 1.

Table 1: Skill representation

Robot	ψ_r		
	Griping	Vision	Pushing
r_1	0	0	1
r_2	0	1	0
r_3	1	1	1
r_4	1	0	0

Definition 3.2. (**Cooperative object transportation task**) A task τ is specified by a 4-tuple $\tau = \langle v, l, t, \Psi \rangle$ where, v is the name of a task (e.g., transporting a box B from location l to location l', lift desk D) etc., where $l \in L$ is the location where the task is arrived/detected, t is the time at which the task arrived, and Ψ is the skill vector required to execute the task.

When a robot detects a task τ, it acquires the information of the skill vector $\Psi_\tau = [\beta_1, \ldots, \beta_p]$ needed to execute the task. The value of a particular skill β_i may be 1 or 0 depending on whether or not a particular skill is required. For example, if $\Psi_\tau = [\beta_1, \ldots, \beta_p]$ = $[0\ 1\ 1]$, this means that the skills β_2 and β_3 are required. Now certain questions that may arise are:

- Is there any robot that can execute a task τ individually?
- Is there any robot who can be a member of team (of two or more robots) to execute the task τ?
- When can we say that a team (of two or more robots) can achieve the task τ?

We introduce two binary operators $\odot$ and $\oplus$ that evaluate to true/false. Let v_1 and v_2 be the skill vectors of a robot and task respectively. The intuitive meaning of the operators are: $v_1 \oplus v_2$ holds, when a robot possess at least one skill for a given task; which signifies that a robot can be a member of a team to execute the task; $v_1 \odot v_2$ holds, when a robot possesses all the skills and possibly more for a given task. (see Definition 3.3). The use of the operator $\odot$ is to check whether a skill vector (from one or multiple robots) has sufficient skills to execute a task, and the operator $\oplus$ checks if a robot can participate in the execution of a task. A robot can participate in a task execution if it has at least one skill required to execute the task.

It is easy to see that r_3 is a suitable candidate since it has all the skills required in τ, expressed by the formula $\psi_{r_3} \odot \Psi_\tau$ which evaluates to true, but $\psi_{r_4} \odot \Psi_\tau$ evaluates to false since r_4 has none of the skills required in Ψ_τ. The robots r_1 and r_2 jointly are suitable candidates, since they each have one of the skills required in Ψ_τ. Thus, a team of r_1 and r_2 can execute τ, since by combining the skills of r_1 and r_2, the skills required for τ, i.e., $\Psi_\tau\ [0\ 1\ 1]$ is obtained. In order to combine the skills of r_1 and r_2, we perform a logical *OR* on the corresponding components of the two vectors; thus we get $\psi_{(r_1,r_2)} = (\psi_{r_1} \vee \psi_{r_2}) = [0\ 1\ 1]$ and $\psi_{(r_1,r_2)} \odot \Psi_\tau$ evaluates to true.

Definition 3.3. (**Skill vector**) Let v_1, v_2 be two p-dimensional skill vectors, $v_1 = [\alpha_1, \ldots, \alpha_p]$, $v_2 = [\beta_1, \ldots, \beta_p]$, where α_i, β_i is either 0 or 1. Let $v_j[i] = \beta_i$ denote the i th component of v_j. We define the following operations on the skill vectors as:

- $v_1 = v_2$ iff $\forall i,\ i \in \{1, \ldots, p\},\ v_1[i] = v_2[i]$
- $v_1 \oplus v_2$ iff $\exists i,\ i \in \{1, \ldots, p\},\ (v_2[i] = 1) \wedge (v_1[i] = 1)$
- $v_1 \odot v_2$ iff $\forall i,\ i \in \{1, \ldots, p\},\ (v_2[i] = 1) \rightarrow (v_1[i] = 1)$
- $v_1\ op\ v_2 = [v_1[1]\ op\ v_2[1], \ldots, v_1[i]\ op\ v_2[i], \ldots, v_1[p]\ op\ v_2[p]]$, where $op \in \{\vee, \wedge\}$ denotes Boolean operations on the vectors.
- $op\ (v_1, \ldots, v_n) = (...((v_1\ op\ v_2)\ op\ v_3) \ldots\ op\ v_n)$.

Definition 3.4. (**Skill vector of a coalition**) Let $\Gamma = \{x_1, \ldots, x_k\}$ be a coalition of k robots, where $x_1, \ldots, x_k$ are the variables and they are place-holders for $r_1, \ldots, r_n$. We define a p-dimensional skill vector of the coalition Γ as $\Upsilon_\Gamma = \vee(\psi_{x_1}, \ldots, \psi_{x_k})$.

Example: Suppose three robots r_1, r_2 and r_3 having 4-dimensional skill vector $\psi_{r_1} = [1, 1, 0, 0]$, $\psi_{r_2} = [0, 0, 0, 1]$ and $\psi_{r_3} = [1, 1, 0, 1]$ respectively are chosen for coalition. The skill vector of the coalition Υ_Γ ($\Gamma = \{r_1, r_2, r_3\}$) is calculated by doing the point-wise logical OR of corresponding skills of the robots (r_1, r_2, r_3), shown in Table 2.

Table 2: Calculation of the skill vector of coalition Γ

Skill vector	α_1	α_2	α_3	α_4
ψ_{r_1}	1	1	0	0
ψ_{r_2}	0	0	0	1
ψ_{r_3}	1	1	0	1
$\Upsilon_\Gamma = \vee(\psi_{x_1}, \ldots, \psi_{x_k})$	1	1	0	1

Definition 3.5. (**When can a coalition Γ execute a task τ**) A coalition Γ of k robots can perform a task τ with vector Ψ_τ if $\Upsilon_\Gamma \odot \Psi_\tau$ holds.

Definition 3.6. (**Cost of a task execution**) Let $\Gamma = \{x_1, \ldots, x_n\}$ be a team that can execute a task $\tau = \langle v, l, t, \Psi \rangle$ where each member of the team was located at $loc(x_i, t')$, $t' > t$. The cost of a team Γ for executing τ is $C_{\langle \Gamma, \tau \rangle} = \sum_{x_i \in \Gamma} \mu_{\langle x_i, \tau \rangle}$ where

$$\mu_{\langle x_i, \tau \rangle} = \mathbf{p}(x_i, \tau) \times \frac{1}{\theta_{x_i}} \; + \; \mathbf{d}(loc(x_i, t'), l) \times \theta'_{x_i}$$

where, $\theta_{x_i}, \theta'_{x_i} \in (0, 1]$ denote remaining battery backup and battery consumption rate respectively of x_i, $\mathbf{p}(x_i, \tau)$ is the basic price of x_i for τ, $\mathbf{d}(l_1, l_2)$ is the distance covered when moving from l_1 to l_2. Basic price is the minimum cost that the each robot levy like *Uber* or *Ola* charge some basic (minimum) fare for a journey. A robot with higher θ ensures that it will not fail due to its more remaining battery backup. A robot with lower θ' ensures that it will last for a longer period of time.

Definition 3.7. **(Dominant coalition)** Let $\Gamma_1, \ldots, \Gamma_{n'}$ be the possible coalitions for executing a task τ. We call Γ_i a dominant coalition if $C_{\langle \Gamma_i, \tau \rangle} \leqslant C_{\langle \Gamma_j, \tau \rangle} \; \forall \; j \in \; \{1, \ldots, n'\} \backslash \{i\}$.

Definition 3.8. **(Mission)** A mission consists of a finite set of transportation tasks M. A mission can be accomplished if for each $\tau_i \in M$, a dominant coalition Γ_i is found.

4 PROPOSED APPROACH

We make the following assumptions to design a distributed algorithm for solving the problem of cooperative object transportation. Multiple autonomous robots are working in a dynamic environment, and a robot can execute at most one task at a time. Each robot has a unique identifier *id*. We consider a wireless network that is lossless, message delay is finite, data is not corrupted during transmission. Messages are delivered in a FIFO (first-in-first-out) manner.

The informal description of the proposed distributed algorithm is as follows. Each robot may be in any one of the following states IDLE, ANALYZE, PROMISE, or BUSY at any instant of time. Let a robot r, in IDLE state, detects a task $\tau = \langle v, l, t, \Psi \rangle$, where $\psi_r \oplus \Psi\tau$ holds. On detection of the task τ, r analyzes whether or not it can execute the task alone. If r cannot execute the task alone, ($\psi_r \odot \Psi_\tau$ evaluates to false), a team has to be formed.

To form a team for the execution of a task, a robot i communicates with other robots. We refer to i as an initiator, and the other robots as non-initiators. The overall coalition formation process for initiator i is given in Algorithm 1 and that for non-initiator j is given in Algorithm 2. To form a coalition, an initiator i sends a Request message whose format is $\langle id_i, v_\tau, l_\tau, (\Psi_\tau - \psi_i) \rangle$, moves to *ANALYZE* state, and waits for some time, say Δ. Here, $(\Psi_\tau - \psi_i)$ denotes the remaining skills required for the task.

It is assumed that a message would be delivered only to the robots present in the environment at that moment in time. A non-initiator j, who has at least one skill required to execute the task (i.e., $\psi_j \oplus \Psi_\tau$), will send a Willing (whose format is $\langle id_j, \psi_j, \mathbf{p}_j, \theta'_j, \theta_j, loc(j, t') \rangle$), if it is in IDLE state. The initiator increments its counter c when it receives a Willing message from a non-initiator.

The initiator constructs a matrix A_S$[a_{ij}]$ of dimension $(c + 1) \times p$, to keep the record of skills received via Willing messages. The matrix is initialized with zero, i.e., $a_{ij} = 0$; $\forall i \in n, \forall j \in p$. If a particular skill is available in the non-initiator's skill vector ψ_j, the value of that cell in the matrix A_S[] is updated with 1. The logical OR of each column of the matrix A_S[] is done after receipt of Willing messages. Now the initiator checks whether or not the task τ can be executed by the updated skill vector. If yes, a dominant coalition of size k is selected as per Definition 3.7.

The initiator sends a Confirm message to (k - 1) robots who are selected and it sends Not-required message to (c - (k - 1)) robots who had sent Willing message but are not selected. Now i changes its state from ANALYZE to BUSY or IDLE depending on whether or not a coalition is formed. The matrix construction and updation is illustrated with an example later in this section.

The computations are done based on the current state that may be IDLE (line no. 1-8), PROMISE (line no. 9-18), BUSY (line no. 20-24) in Algorithm 2. Within a state, the type of message is checked and appropriate actions are taken. For example, when state is IDLE, if a Request message is received, it becomes PROMISE. Now the robot sends a Willing message to the initiator. When a robot goes to BUSY state, it means task execution would commence soon.

Algorithm 1: Distributed Algorithm: Initiator i

robot i detects a task $\tau = \langle v, l, t, \Psi \rangle$;
state := ANALYZE;
broadcast Request;
create a matrix A_S$[a_{ij}]_{n \times p}$; ◇ to keep the record of information received
start timer and wait for Δ unit of time;
$c := 0$; ◇ to count the number of Willing messages
case *msg* =Willing **do**
    $c := c + 1$;
    update matrix A_S$[a_{ij}]_{n \times p}$;
end
if *(τ can be executed with availbale skill vector)* **then**
    for each possible coalition, calculate cost as per Definition 3.6;
    select the dominant coalition, say Γ of size k as per the Definition 3.7;
    send Confirm to (k - 1) members of Γ;
    send Not-Required to (c - (k - 1)) non-initiators;
    state := BUSY;
    initiate execution of τ when (k - 1) members of Γ arrive at l;
else
    send Not-Required to c non-initiators;
    state := IDLE;
end
case *msg* = Request **do**
    skip;
end
if *(*Willing *message is received in a state* $\neq$ ANALYZE *from j)* **then**
    send Not-Required to j;
end

In PROMISE state, non-initiator receives either a Confirm or a Not-required message. Upon receipt of a Confirm message, it approaches the location of the task (Algorithm 2, line no. 9-12) and changes its state to BUSY. If the non-initiator receives Not-required message, then it changes its state from PROMISE to IDLE (Algorithm 2, line no. 13-15). The non-initiator who receives the Confirm message and moves towards initiator location, sends a beacon signal to acknowledge that it has reached. When all selected robots reach the location of the task, they synchronize and align themselves to be ready for task execution. Eventually, initiator commands for

Algorithm 2: Distributed Algorithm: Non-initiator j

```
case state = IDLE do
    case msg = Request and ψ_j ⊕ Ψ_τ do
        state := PROMISE; send Willing to i;
    end
    case state = IDLE and ¬(ψ_j ⊕ Ψ_τ) do
        skip;
    end
end
case state = PROMISE do
    case msg = Confirm do
        state := BUSY; move to l;
    end
    case msg = Not-Required do
        state := IDLE;
    end
    case msg = Request do
        skip;
    end
end
case state = BUSY do
    case msg = Request do
        skip;
    end
end
```

task execution and they jointly execute the task. When the task execution has been completed, the coalition is dissolved and the members of the coalition change their state to IDLE. In this way, all the robots of the environment behave according to Algorithms 1 and 2 to accomplish the mission.

We have assumed that communication is lossless. If each non-initiator has a timer, so like the initiator, it would also timeout and disassociate itself from the coalition formation process after the set span of time. In this way, the algorithm can be made tolerant to message loss.

4.1 Illustration of coalition formation

To better understand the working of the proposed distributed algorithm, let us consider the following example. Suppose that there are five robots, $r_1, \ldots, r_5$, present in a dynamic environment, and the mission is to execute tasks that require 6-dimensional skill vector. Initially all the robots are in IDLE state. The skill vector ψ, basic price $\mathbf{p}$, remaining battery backup θ', and battery consumption rate θ for each robot are given in Table 3.

Table 3: ψ, basic price p, θ' and θ of robots

Robot	ψ	p	θ'	$\theta\%$
r_1	[1, 0, 0, 0, 0, 0]	10	0.50	60
r_2	[1, 0, 1, 0, 0, 0]	40	0 .50	80
r_3	[0, 0, 0, 1, 0, 1]	30	0.50	80
r_4	[0, 1, 1, 0, 0, 0]	50	0.70	75
r_5	[0, 1, 1, 1, 0, 0]	98	0.90	72

Let at some moment in time, r_1 detects a task τ_1 that requires skill vector $\Psi_\tau = [1, 1, 0, 1, 0, 0]$ and battery backup $\geqslant$ 40%. It is clear that r_1 cannot execute the task alone as it does not possesses sufficient skills to execute the task. Hence, r_1 starts the coalition formation process by broadcasting a Request message for the remaining skills, $\Psi' = ([1,1,0,1,0,0] - [1,0,0,0,0,0] = [0,1,0,1,0,0])$. Then r_1 changes its state to ANALYZE (see Figure 1a).

The robots r_3, r_4, and r_5 send Willing message to the initiator, r_1 as they are in IDLE state and have at least one skill required to execute the task. r_2 ignores the Request message in IDLE state since it does possesses any required skill (see Figure 1b). The Willing message contains the *id*, skill vector, basic price **p**, battery consumption rate θ', remaining battery θ, and location l of the non-initiator. The information received via Willing messages are recorded in Table 4. Then r_1 calculates the distance of the non-initiators from their locations to the location of the task.

Table 4: The information received via Willing **messages**

Robot	ψ	p	θ'	θ	d
r_3	$[0, 0, 0, 1, 0, 1]$	30	0.5	0.80	12 unit
r_4	$[0, 1, 1, 0, 0, 0]$	50	0.7	0.75	15 unit
r_5	$[0, 1, 1, 1, 0, 0]$	98	0.9	0.72	10 unit

r_1 updates the availability matrix on the basis of values received in Willing messages. In this case the value of c is 3. The updated matrix A_S[] is given Table 5.

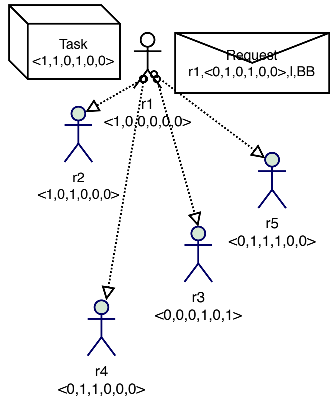
Table 5: Updated available matrix: A_S[]

Robot	α_1	α_2	α_3	α_4	α_5	α_6
r_1	1	0	0	0	0	0
r_3	0	0	0	1	0	1
r_4	0	1	1	0	0	0
r_5	0	1	1	1	0	0
Cur_Avail	1	1	1	1	0	1

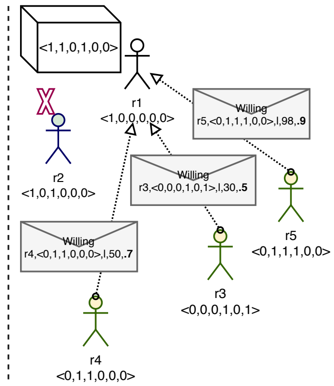
Now, r_1 checks whether the combined skills of r_1, r_3, r_4, and r_5 would satisfy the skill vector of the task (([1,1,1,1,0,1] $\odot$ [1,1,0,1,0,0]) holds). Now we have to find the dominant coalition. The cost calculation for each robot is given below.
$\mu_{\langle r_1,\tau\rangle} = 10 \times \frac{1}{.60} + .5 \times 0 = 16.67$ unit; $\mu_{\langle r_3,\tau\rangle} = 30 \times \frac{1}{.80} + 0.5 \times 12 = 43.50$ unit; $\mu_{\langle r_4,\tau\rangle} = 50 \times \frac{1}{.75} + 0.7 \times 15 = 77.17$ unit; $\mu_{\langle r_5,\tau\rangle} = 98 \times \frac{1}{.72} + 0.9 \times 10 = 145.11$ unit; From Table 5, four possible coalitions can execute the task; $\Gamma_1 = \{r_1, r_3, r_4\}$, $\Gamma_2 = \{r_1, r_4, r_5\}$, $\Gamma_3 = \{r_1, r_3, r_5\}$, and $\Gamma_4 = \{r_1, r_5\}$.

Cost of each coalition: $\text{cost}_{\langle\Gamma_1,\tau\rangle}$ = 16.67 + 43.50 + 77.17 = **137.34** unit; $\text{cost}_{\langle\Gamma_2,\tau\rangle}$ = 16.67 + 77.17 + 145.11 = 238.95 unit; $\text{cost}_{\langle\Gamma_3,\tau\rangle}$ = 16.67 + 43.50 + 145.11 = 205.28 unit; $\text{cost}_{\langle\Gamma_4,\tau\rangle}$ = 16.67 + 145.11 = 161.78 unit.

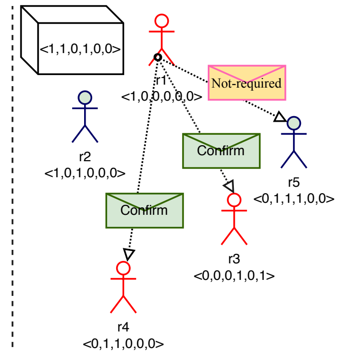
Thus, Γ_1 having robots r_1, r_3 and r_4, is the dominant coalition since it has the minimum cost. The total number of robots in the coalition is 3. Then r_1 sends Confirm message to r_3 and r_4 and Not-required message to r_5 (see Figure 1c). On receipt of Confirm message, r_3 and r_4 change their state to BUSY and start moving towards the location of task; r_5 changes its state to IDLE. When r_3 and r_4 reach the location of the task, then all of them start the execution of the task. This example illustrates that a minimum sized coalition need not be a dominant coalition (in this case Γ_4).



(a) r_1 **broadcast the** Request



(b) r_1 **receives** Willing **messages from non-initiator**



(c) r_1 **sends** Confirm **and** Not-required **messages**

Figure 1: Illustration of coalition formation

5 EXPERIMENTAL RESULTS

The proposed distributed algorithm designed to accomplish a mission by autonomous robots in a dynamic environment is simulated and tested with ARGoS [15]. The performance of the algorithm is recorded and analyzed.

5.1 An object transportation scenario

We consider an object transportation scenario in an urban search and rescue environment [13], that provides a platform for disaster management where heterogeneous field agents (fire brigade agents, ambulance agents, and police force agents) collaborate with each other to manage a mimicked calamity situation. The agents are responsible for extinguishing the fire, save lives of human beings, and transport obstacles from the road. The role of the police force agent is crucial, as they clear the blocked roads and thereby making an easy passage for ambulance agents and fire brigade agents to perform their tasks. After a disaster, a road in the city may get blocked by any type of obstacle (road-blockers), i.e., light or heavy. The obstacles may be scattered randomly on the road. The road clearance problem in search and rescue simulation environment requires a distributed approach (as given in Section 4) that guarantees road-clearance, regardless of the possibility of heavy obstacles present on the road.

In the simulated environment, 5 robots and 4 different types of tasks (transportation of obstacles) are present. The 6-dimensional skill vectors required for executing the tasks are: Ψ_{τ_1}=[1,0,1,1,0,1], Ψ_{τ_2}=[0,0,0,0,1,1], Ψ_{τ_3}=[1,0,0,0,0,1], and Ψ_{τ_4}=[1,0,0,0,0,0] (see Table 6). The tasks τ_1, τ_2, τ_3 and τ_4 are represented with large disc, large box, small disc and small box respectively. The battery backup required for executing each task is considered to be $\geqslant$ 50%.

Table 6: The value of skill vector Ψ required for each task

Task	Skill vector
τ_1	[1, 0, 1, 1, 0, 1]
τ_2	[0, 0, 0, 0, 1, 1]
τ_3	[1, 0, 0, 0, 0, 1]
τ_4	[1, 0, 0, 0, 0, 0]

The skill vector ψ, basic price **p** and battery consumption coefficient θ' of each robot presented in the environment is given in Table 7. For the sake of implementation, we consider one basic price for one robot instead of a different basic price for each skill of the robot. Initial battery backup θ of each robot is assumed to be 100%.

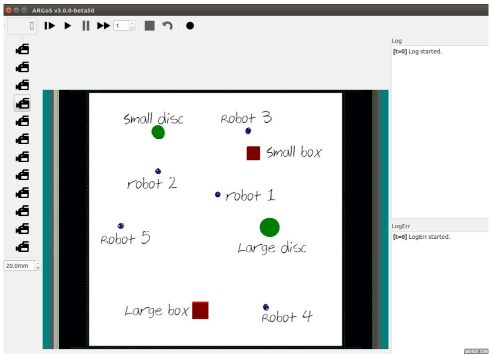
Table 7: The value of skill vector ψ, basic price p, and battery consumption coefficient θ' of each robot

Robot	Skill vector (ψ_r)	**p**	θ'
r_1	[1, 0, 0, 0, 0, 0]	40	0.5
r_2	[0, 0, 0, 0, 0, 1]	50	0.6
r_3	[1, 1, 0, 0, 0, 0]	70	0.7
r_4	[0, 0, 0, 0, 1, 1]	70	0.8
r_5	[0, 0, 1, 1, 0, 0]	60	0.6

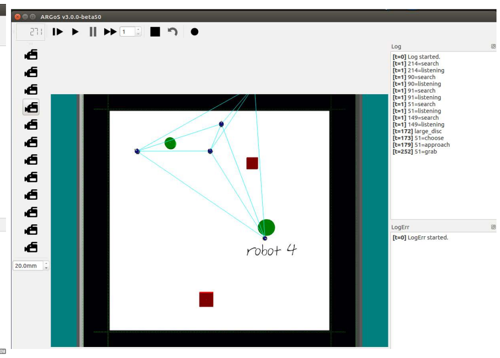
The initial snapshot of the environment is shown in Figure 2a. The robots explore the white space to find obstacles. The r_4 detects the large disc (see Figure 2b) and form the coalition. The coalition members reach the location of the obstacle (Figure 2b and 2c). Then all the members grab the obstacle and orient themselves appropriately (see Figure 2d). The robots then move the obstacle towards the black space. Eventually, the obstacle is placed in the black space (see Figure 2e) and thereafter the robots once again start exploring the white space for other obstacles and move them as well (see Figure 2f).

5.2 Results and Analysis

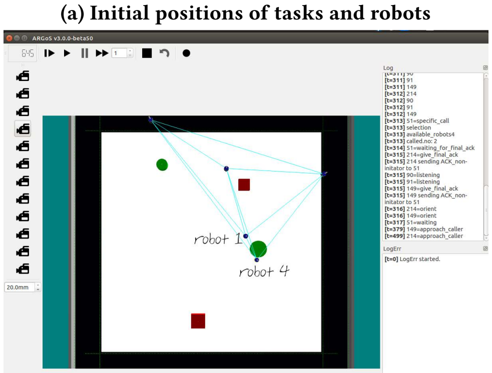
The experimental results obtained are shown in Tables 8, 9, 10, 11, and 12. The results clearly indicate that the coalition structure cannot be determined a priori since, in a dynamic environment, the states and roles of the robots change with time. It also shows that the number of robots for a task execution cannot be fixed in advance, rather it is decided at runtime. For example, when coalition formation for the task τ_1 is invoked by different initiators in different experiments, it may result in different coalitions. Each experiment is conducted with a new set of robots present in the environment, new states and locations of the robots that are distributed randomly



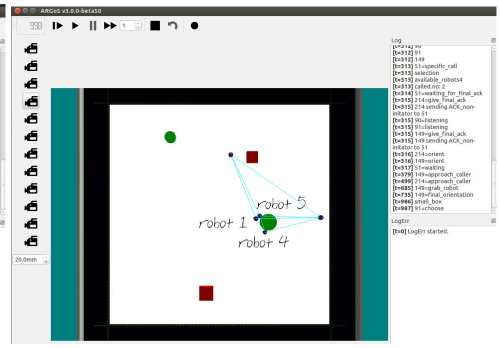
(a) Initial positions of tasks and robots



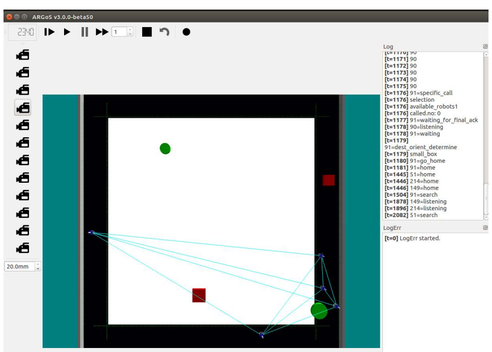
(b) r_4 detects a task (large disc)



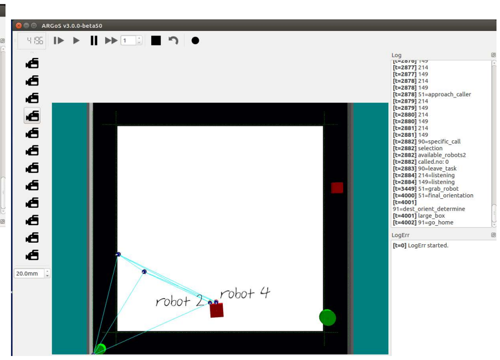
(c) r_1 moving towards large disc



(d) r_4, r_1, r_5 grabbing the large disc



(e) Large disc is placed at black space and coalition is dissolved



(f) r_2 and r_4, start moving large box

Figure 2: Illustration of task detection, coalition formation and its execution by the robots of the coalition

in the white space. The obstacles are also distributed randomly in white space.

Task τ_1 is initialized by r_3 in experiment 1 (see Table 8), by r_2 in experiments 2 and 3 (see Tables 9, 10), and by r_4 in experiments 4, and 5 (see Tables 11, 12). Moreover, the same task τ_1 is executed by different coalitions in different experimental runs. Task τ_1 is executed by a coalition having robots $\{r_3, r_4, r_5\}$ in experiment 1, $\{r_2, r_1, r_5\}$ in experiment 2, $\{r_4, r_1, r_5\}$ in experiment 4. It is clear from these results that the initiator, as well as members of the coalition, are decided at runtime in the dynamic environment.

Another important point is that the same task τ_3 is executed by the different number of robots in different experiments. The task τ_3 is initialized by robot r_2 in experiment 3 (Table 10) and executed by $\{r_2, r_4\}$ while, the same task τ_3 is initiated and executed by a single robot r_4 in experiment 5 and r_2 in experiment 6 (see Table 12). The results indicate that the number of robots required to execute a task is not known a priori in the dynamic environment, and it is decided at only runtime. Also, the cost of each task execution varies depending on the states and locations of the robots. Time taken to complete the goal also varies but remains in a limit. Similar pattern is shown in the execution of tasks τ_2 and τ_4. In all the experiments, each task is executed by a dominant coalition, i.e., minimum cost.

Table 8: Experiment 1

Tasks	Initiator for the task	Coalition	Cost
τ_1	r_3	$\{r_3, r_4, r_5\}$	1006.07
τ_2	r_2	$\{r_2, r_4\}$	862.02
τ_3	r_1	$\{r_1, r_2\}$	295.18
τ_4	r_1	$\{r_1\}$	40.00
Total_Cost			**2203.27**
Total_Time			**45.99**

Table 9: Experiment 2

Tasks	Initiator for the task	Coalition	Cost
τ_1	r_2	$\{r_2, r_1, r_5\}$	818.72
τ_2	r_4	$\{r_4\}$	70.00
τ_3	r_3	$\{r_3, r_4\}$	717.11
τ_4	r_1	$\{r_1\}$	40.00
Total_Cost			**1645.83**
Total_Time			**46.16**

Table 10: Experiment 3

Tasks	Initiator for the task	Coalition	Cost
τ_1	r_2	$\{r_2, r_1, r_5\}$	615.00
τ_2	r_2	$\{r_2, r_4\}$	421.20
τ_3	r_3	$\{r_3, r_2\}$	1222.10
τ_4	r_3	$\{r_3\}$	70.00
Total_Cost			**2328.3**
Total_Time			**72.8**

Table 11: Experiment 4

Tasks	Initiator for the task	Coalition	Cost
τ_1	r_4	$\{r_4, r_1, r_5\}$	1163.28
τ_2	r_2	$\{r_4, r_2\}$	1241.49
τ_3	r_3	$\{r_2, r_3\}$	163.85
τ_4	r_3	$\{r_3\}$	70.00
Total_Cost			**2638.62**
Total_Time			**75.83**

Table 12: Experiment 5

Tasks	Initiator for the task	Coalition	Cost
τ_1	r_4	$\{r_4, r_1, r_5\}$	1460.52
τ_2	r_4	$\{r_4\}$	70.00
τ_3	r_3	$\{r_3, r_4\}$	1143.72
τ_4	r_3	$\{r_3\}$	70.00
Total_Cost			**2724.24**
Total_Time			**51.71**

6 CONCLUSION

In this paper, we considered the problem of cooperative object transportation in a generic dynamic environment, where the location and time of object arrival, the number of robots required to transport the object, states, and location of the robots are not known in advance. A formal framework for cooperative task execution is provided. We proposed a distributed approach where the robots communicate among themselves by exchanging messages without any human or central intervention to form a coalition for task execution. A prototype model of the proposed algorithm is developed in ARGoS; a multi-robot simulation environment, and it was tested by performing each experiment 10 times. The simulation results are quite encouraging and it demonstrates how robots are executing the tasks and that different coalitions are formed in different simulation runs for the same task. As part of our future work, we wish to implement the prototype model on real robots and test the efficiency of the system.

ACKNOWLEDGMENT

The authors thank the anonymous referees for their valuable comments that were helpful for improving the paper. The third author was in part supported by a research grant from Google.

REFERENCES

[1] S. et al. Adinandra. 2012. Flexible transportation in warehouses. In *Automation in Warehouse Development*. Springer, 191–207. https://doi.org/10.1007/978-0-85729-968-0_14

[2] Muhanad H Mohammed Alkilabi, Aparajit Narayan, and Elio Tuci. 2017. Cooperative object transport with a swarm of e-puck robots: robustness and scalability of evolved collective strategies. *Swarm Intelligence* 11, 3-4 (2017), 185–209. https://doi.org/10.1007/s11721-017-0135-8

[3] J. et al. Chen. 2015. Occlusion-based cooperative transport with a swarm of miniature mobile robots. *IEEE Transactions on Robotics* 31, 2 (2015), 307–321. https://doi.org/10.1109/tro.2015.2400731

[4] Micael S Couceiro, David Portugal, and Rui P Rocha. 2013. A collective robotic architecture in search and rescue scenarios. In *ACM Symposium on Applied Computing*. ACM, 64–69. https://doi.org/10.1145/2480362.2480377

[5] Brian P Gerkey and Maja J Mataric. 2002. Sold!: Auction methods for multirobot coordination. *IEEE transactions on robotics and automation* 18, 5 (2002), 758–768. https://doi.org/10.1109/tra.2002.803462

[6] Brian P Gerkey and Maja J Matarić. 2004. A formal analysis and taxonomy of task allocation in multi-robot systems. *The International Journal of Robotics Research* 23, 9 (2004), 939–954. https://doi.org/10.1177/0278364904045564

[7] Tyler Gunn and John Anderson. 2015. Dynamic heterogeneous team formation for robotic urban search and rescue. *J. Comput. System Sci.* 81, 3 (2015), 553–567. https://doi.org/10.1016/j.procs.2013.06.009

[8] J. et al. Gutiérrez. 2016. The multiple team formation problem using sociometry. *Computers & Operations Research* 75 (2016), 150–162. https://doi.org/10.1016/j.cor.2016.05.012

[9] G. et al. Habibi. 2015. Distributed centroid estimation and motion controllers for collective transport by multi-robot systems. In *ICRA*. IEEE, 1282–1288.

[10] Yan Kong, Minjie Zhang, and Dayong Ye. 2015. An auction-based approach for group task allocation in an open network environment. *Comput. J.* 59, 3 (2015), 403–422. https://doi.org/10.1093/comjnl/bxv061

[11] C Ronald Kube and Hong Zhang. 1993. Collective robotics: From social insects to robots. *Adaptive behavior* 2, 2 (1993), 189–218. https://doi.org/10.1177/105971239300200204

[12] Theodoros Lappas, Kun Liu, and Evimaria Terzi. 2009. Finding a team of experts in social networks. In *ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 467–476. https://doi.org/10.1109/icdmw.2012.54

[13] Yugang Liu and Goldie Nejat. 2013. Robotic urban search and rescue: A survey from the control perspective. *Journal of Intelligent & Robotic Systems* 72, 2 (2013), 147–165. https://doi.org/10.1007/s10846-013-9822-x

[14] Maja J Mataric, Martin Nilsson, and Kristian T Simsarin. 1995. Cooperative multi-robot box-pushing. In *International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, Vol. 3. IEEE, 556–561. https://doi.org/10.1109/iros.1995.525940

[15] C. et al. Pinciroli. 2012. ARGoS: a modular, parallel, multi-engine simulator for multi-robot systems. *Swarm intelligence* 6, 4 (2012), 271–295.

[16] Erol Şahin. 2004. Swarm robotics: From sources of inspiration to domains of application. In *International workshop on swarm robotics*. Springer, 10–20. https://doi.org/10.1007/978-3-540-30552-1_2

[17] Elio Tuci, Muhanad H Alkilabi, and Otar Akanyeti. 2018. Cooperative object transport in multi-robot systems: A review of the state-of-the-art. *Frontiers in Robotics and AI* 5 (2018), 59. https://doi.org/10.3389/frobt.2018.00059

[18] I. et.al. Vasilyev. 2015. Use of mobile robots groups for rescue missions in extreme climatic conditions. *Procedia Engineering* 100 (2015), 1242–1246. https://doi.org/10.1016/j.proeng.2015.01.489

[19] Ying Wang and Clarence W De Silva. 2006. Multi-robot box-pushing: Single-agent q-learning vs. team q-learning. In *International Conference on Intelligent Robots and Systems*. IEEE, 3694–3699. https://doi.org/10.1109/iros.2006.281729

[20] Bing Xie, Shaofei Chen, Jing Chen, and LinCheng Shen. 2018. A mutual-selecting market-based mechanism for dynamic coalition formation. *International Journal of Advanced Robotic Systems* 15, 1 (2018), 1729881418755840. https://doi.org/10.1177/1729881418755840

[21] Zhi Yan, Nicolas Jouandeau, and Arab Ali Cherif. 2013. A survey and analysis of multi-robot coordination. *International Journal of Advanced Robotic Systems* 10, 12 (2013), 399. https://doi.org/10.5772/57313