Ad Hoc Network-Based Task Allocation With Resource-Aware Cost Generation for Multirobot Systems

Dong-Hyun Lee, Sheir Afgen Zaheer, and Jong-Hwan Kim, *Fellow, IEEE*

***Abstract*—The objective of multirobot task allocation (MRTA) is to assign tasks to robots to minimize the overall cost of task performance. This paper proposes a decentralized MRTA approach considering the robots' residual expendable resources and their limited communication ranges. The proposed approach consists of two algorithms, namely, resource-aware cost generation (RCG) and ad hoc network-based task allocation (ANTA). The RCG algorithm allows each robot to generate a credible cost of task performance considering its residual resources in task planning. The ANTA algorithm constructs the minimal spanning tree network among the robots and determines the robot with the lowest cost for the task through multihop communication in a decentralized manner. The advantages of the proposed approach are minimization of unnecessary task performance cost caused by resource shortage of robots during task execution and the use of an ad hoc network among the robots to allow more robots to participate in the task allocation process. The effectiveness of the proposed approach is demonstrated through computer simulations for multirobot foraging as a test problem.**

***Index Terms*—Ad hoc network-based task allocation (ANTA), decentralized task allocation, multirobot systems, resource-aware cost generation (RCG).**

I. Introduction

MULTIROBOT systems have been widely applied in many different areas that are dangerous, difficult, and expensive for humans [1]. These application domains include exploration [2], [3], vehicle formation control [4]–[6], cooperative surveillance [7], and target tracking [8]. The multirobot task allocation (MRTA) problem is one of the important research areas in multirobot systems. The MRTA problem can be viewed as an optimization problem in which tasks are allocated in a way that minimizes the total task performance cost or maximizes the total utility [1], [9].

This paper is motivated by the MRTA problem for a long-duration mission in which tasks continuously occur at arbitrary locations. In the mission, a team of robots with limited communication range negotiate with each other for task allocation and autonomously execute the tasks as fast as possible, as well as occasionally refill their resources in refill stations. Examples of such application areas are planetary exploration [10], automated warehouse management [11], [12], urban hygiene management [13], and agriculture automation [14]. One of the important issues for such domains is the consideration of the robots' finite expendable resources. The robot resources in this paper refer to any kind of expendable supplies that robots consume during task execution. Examples of such resources include batteries for surveillance robots [15], dust filters for cleaning robots [16], herbicide for weeder robots [17], and snacks for refreshment delivery robots [18]. If the resources are not considered in task allocation, the tasks might be allocated to robots that will run out of resources in the middle of task execution. Since robots with insufficient resources should stop performing the tasks and visit resource stations to refill, the cost generation without considering the robot resources might cause extra expenses, including an increase in task completion time and extra resource consumption.

There have been various studies on multirobot coordination considering robot resources. In [19], each robot uses offline temporal interval propagation to decide which task to execute and when to execute it considering its remaining resources. However, it assumes that all information of tasks is given *a priori* and that the information needed for each robot to make a decision is available. The hybrid planning/learning system developed in [20] schedules a group of robots for a heterogeneous stream of tasks. It applies a reinforcement learning model to obtain a value function to consider positioning of robots in readiness for new tasks and preserving resources. However, since it requires fully connected communication for the robots to form joint plans, it suffers from a scalability problem. In [21] and [22], energy-aware bid generation methods for market-based task allocation are presented. They define the cost as the estimated energy consumption for a robot to complete a task such that the task is assigned to the robot that can perform the task with the lowest energy consumption. However, the approach in [22] assumes that all the tasks should be completed by the robots with initially given energy resources such that it is not applicable for a long-duration mission in which each robot must recharge its resources periodically. In the case of the approaches presented in [22] and [23], the robot that has a lower energy level than a specified threshold autonomously recharges its battery by visiting a charging station. However, they only considered a single resource, i.e., energy, and did not take into

Manuscript received May 28, 2013; revised March 4, 2014; accepted April 23, 2014. Date of publication May 29, 2014; date of current version September 12, 2014. This work was supported by the Ministry of Trade, Industry and Energy of Korea under the Technology Innovation Program supervised by the Korea Evaluation Institute of Industrial Technology, Development of Robot Task Intelligence Technology, under Grant 10045252.

The authors are with the Department of Electrical Engineering, Korea Advanced Institute of Science and Technology, Daejeon 305-701, Korea (e-mail: dhlee@rit.kaist.ac.kr; sheir@rit.kaist.ac.kr; johkim@rit.kaist.ac.kr).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIE.2014.2326987

account the measurement noise and the prediction uncertainty while estimating the robot's resource level.

In addition to the robot resources, limited communication range of the robot is also an important factor in MRTA. The auction methods have been shown to efficiently produce suboptimal solutions with limited communication range [23]–[26]. In an auction approach, any robot that tries to allocate a task becomes an auctioneer and broadcasts an auction call for the task. The others that are interested in the auctioned task become bidders and submit their costs to the auctioneer. The auctioneer then allocates the task to the bidder that has the lowest cost. However, only the direct neighbors of the auctioneer, i.e., the robots that are within a single-hop communication range of the auctioneer, can participate in the task allocation process. Thus, the auction methods might suffer from a misassignment problem if none of the robots in a single-hop distance is capable of performing the task. The probability of misassignment increases as the heterogeneity among the robots increases and/or the robot density within the communication range decreases. To complement the robot's limited communication range, sensor-network-based MRTA approaches have been developed in [27] and [28]. Using a sensor network, the robots that are not within communication range of each other can share their information. However, a static sensor network should be predeployed in the task environment. Its application is hence limited to prestructured environments with a large number of static sensors. Instead of using sensor networks, the approaches in [29] and [30] employ a multihop messaging protocol for task assignment. However, they are limited to homogeneous swarm robots and require the information of the diameter of the network. Moreover, the desired robot assignment rates for the fixed number of tasks should be given to the robots in advance.

In order to address the aforementioned two practical issues in the MRTA problem, this paper proposes two algorithms, namely, a resource-aware cost generation (RCG) algorithm and an ad hoc network-based task allocation (ANTA) algorithm, for efficient decentralized task allocation. The RCG algorithm models multiple resources of a robot as a multivariate normal distribution and utilizes them to estimate the cost of the path consisting of the refill stations that the robot should visit before the task to avoid resource depletion while performing the task. Since the algorithm estimates the residual resources after completing the task, it is capable of minimizing extra resource consumption and time delay due to detour of the robots to refill stations for refilling their resources during task execution. Moreover, unlike the approaches in [21]–[23], where robots do not participate in the task allocation process when they do not have enough energy resources, the RCG algorithm allows the robots to participate in the task allocation process regardless of their residual resources since it is capable of selecting multiple paths consisting of different combinations of refill stations based on their resource levels. On the other hand, the ANTA algorithm constructs the minimal spanning tree and the robots in the network exchange information about the best candidate robot that has the lowest cost for the task. This requires neither perfect communication links among the robots nor infinite bandwidth since the robots in the local network simply share the information about which robot has the lowest cost for the task. Moreover, it is capable of finding a better solution than the auction algorithms with single-hop communication since more robots are allowed to participate in the task allocation process by using the multihop communication network.

This paper is organized as follows. Section II describes the resource-aware task allocation problem and the broadcast tree communication as the background for this paper. Sections III and IV propose the RCG and ANTA algorithms, respectively. In Section V, the performance of the proposed approach is tested using multirobot foraging simulations, and the results are discussed and analyzed. Finally, concluding remarks follow in Section VI.

II. Background

A. Resource-Aware MRTA Problems

The objective of MRTA is to assign tasks to the robots that can minimize the global cost. For a robot set $\mathcal{R}$ ($\mathcal{R} = \{1, \ldots, N_R\}$), where N_R is the number of robots, and a task set $\mathcal{T}$ ($\mathcal{T} = \{1, \ldots, N_T\}$), where N_T is the number of tasks, the MRTA problem can be represented as minimizing the global cost, which is defined as

$$\min \sum_{i \in \mathcal{R}} \sum_{j \in \mathbf{A}^i} c_j^i, \qquad i \in \mathcal{R};\ \mathbf{A}^i \subset \mathcal{T}$$

subject to

$$\mathbf{A}^i \cap \mathbf{A}^k = \emptyset, \qquad i \neq k;\ i, k \in \mathcal{R} \tag{1}$$

where c_j^i is the cost of robot i for task j, and $\mathbf{A}^i$ is the task list of robot i, where its accepted tasks are stored. The constraint in (1) represents that one task should be allocated to a single robot. The cost, in general, depends on the completion time or the travel distance of the robot for the task.

In realistic scenarios where the robots consume their resources during task execution, the task should be allocated to the robot that has enough resources to perform the task with the lowest cost. If the robot resources are not considered in task allocation, the robots might not be able to complete the tasks due to unexpected depletion of their resources during task execution. Moreover, since they should stop performing the tasks and visit resource refill stations to refill their resources, extra cost, such as time delay in task completion or extra resource consumption, is incurred. This implies that an additional constraint is required in (1) to prevent the MRTA algorithm from allocating the tasks to resource-depleted robots. In order to consider the robot resources in task allocation, the resource-aware MRTA problem is defined by adding the resource constraint in (1), which is defined as

$$\sum_{i \in \mathcal{R}} \sum_{j \in \mathbf{A}^i} \sum_{l \in \mathcal{L}} y_{j,l}^i = 0$$

with

$$y_{j,l}^i = \begin{cases} 0, & \text{if } x_{j,l}^i > \gamma_l^i \\ 1, & \text{otherwise} \end{cases} \tag{2}$$

where $\mathcal{L}$ is the resource set ($\mathcal{L} = \{1, \ldots, N_L\}$), N_L is the number of resources, $x_{j,l}^i$ is the lth resource level of robot i after completing task j, and γ_l^i represents the threshold of the

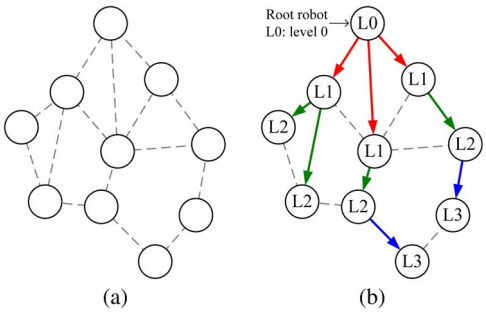


Fig. 1. Illustration of broadcast tree communication. (a) Ad hoc communication network of ten nodes is constructed with limited communication range. (b) Root on the top of the network broadcasts its message to its neighbors, and a minimal spanning tree is constructed as the messages propagate from the root to the other nodes in the network.

lth resource level. The resource constraint represents that all the resource levels of the robot after completing the assigned tasks should be higher than their threshold levels.

One of the most common decentralized approaches to solve the resource-aware MRTA problem is to let each robot estimate its future resource levels after completing the task, assuming that the task is assigned to itself [22], [23]. If all of the robot's resource levels exceed the thresholds, the robot is allowed to participate in the task allocation process. Otherwise, it either waits for a new task that it is capable of fulfilling with its residual resources or goes to a refill station. The RCG algorithm, on the other hand, solves the problem by taking into account robot resources in task planning such that the robot generates the cost for the plan that satisfies the resource constraint in (2). As a result, any robot can participate in the task allocation process regardless of its resource levels. The RCG algorithm is described in detail in Section III.

B. Broadcast Tree Communication

The broadcast tree communication is a multihop messaging procedure used in sensor networks and routing protocols to find routes through ad hoc networks [31]–[33]. It builds a minimal spanning tree, i.e., a directed acyclic message propagation tree, as the message propagates from the root to the leaf nodes. Fig. 1 shows an illustration of broadcast tree communication, where the nodes and the dotted edges represent the robots and the communication links among the robots, respectively. First, the root robot broadcasts a message to its neighbors. The message contains a level, i.e., the number of communication hops from the root. The robot that has received multiple messages of the same type only selects the one that has the lowest level, and it chooses the sender of the selected message as its parent. The robot then increments the level in the selected message by one, sets the incremented level as its level, and broadcasts its message to its neighbors. Since the message that has traveled the least number of hops from the root is selected by the robots in the network, the message propagation produces a directed acyclic spanning tree from the ad hoc network graph G. The total computation of the tree construction requires $O(\text{diam}(G))$ rounds, where $\text{diam}(G)$ is the maximum distance over all vertices in the graph G.

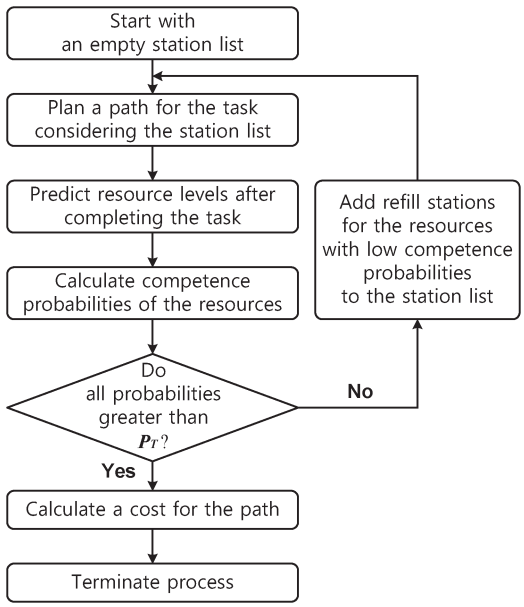


Fig. 2. Flowchart of the RCG algorithm.

In order to form a minimal spanning tree network among the robots with limited communication range, the proposed ANTA algorithm utilizes a modified version of the broadcast tree communication. The ANTA algorithm lets the robots exchange local information about the best candidate robot that has the lowest cost for the task while constructing the tree network. If there is no capable robot in the network, the robots incrementally extend the communication tree to the maximum tree level until finding the capable robot with the minimum cost. The ANTA algorithm is described in detail in Section IV.

III. RCG

A. Algorithm Description

The RCG algorithm is an iterative method that searches for the path that satisfies the resource constraint in (2) with the minimum cost. Each robot in a group calculates its cost using the algorithm and negotiates with other robots to determine the one with the lowest cost. A flowchart of the algorithm is shown in Fig. 2. First, the algorithm plans a path for the task, assuming that the residual resources are enough to complete the task. It then calculates the predicted resource levels that would be left after completing the task by taking the measurement and prediction uncertainties of the resources into account. In order to examine whether the task plan satisfies the resource constraint in a probabilistic manner, the algorithm calculates the competence probability of each resource, which is defined as the probability of the resource being greater than its threshold. If all the competence probabilities are greater than the predefined probability threshold, which is denoted by P_T, the algorithm assumes that the path satisfies the resource constraint. Thus, it calculates the cost for the path and terminates the cost generation process. On the other hand, if any of the competence probabilities is lower than P_T, the algorithm replans a path via the refill stations for resources with low competence probabilities and repeats the process, as shown in Fig. 2. The visiting

order of the refill stations that the robot plans to visit is recorded in a station list. It is used for calculating the predicted resource levels and the cost for the task. As a result, the algorithm allows the robot to plan a path that visits the refill stations for the resources that are likely to be depleted in the middle of task execution and to find the minimum cost of the path that satisfies the resource constraint.

In the RCG algorithm, the array of resource levels, which is defined as the resource vector, is modeled as a multivariate normal distribution of a N_L-dimensional random vector, such that the resource vector of robot i, i.e., $\mathbf{x}^i$, is defined as

$$\mathbf{x}^i = [\, x_1^i \quad x_2^i \quad \dots \quad x_{N_L}^i \,] \sim \mathcal{N}_{N_L}(\boldsymbol{\mu}^i, \Sigma^i) \tag{3}$$

where $\boldsymbol{\mu}^i$ and Σ^i are the mean vector and the covariance of $\mathbf{x}^i$, respectively; and $x_l^i \sim \mathcal{N}(\mu_l^i, {\sigma_l^i}^2)$ is the lth resource level of robot i with mean μ_l^i and standard deviation σ_l^i. The predicted resource vector of robot i for task j, i.e., $\mathbf{x}_j^i$, is defined as

$$\mathbf{x}_j^i = g\left(\mathbf{x}_{\text{start}}^i, \mathbf{s}_j^i\right) + \epsilon_j^i \tag{4}$$

where $\mathbf{x}_{\text{start}}^i$ is the estimated resource vector when the robot starts the task, $\mathbf{s}_j^i$ is the station list of robot i for task j, $g(\mathbf{x}_{\text{start}}^i, \mathbf{s}_j^i)$ is the residual resource prediction function, and ϵ_j^i is the random vector that models the uncertainty introduced by the resource prediction. Its mean is zero, and its covariance is denoted by R_j^i. If the robot does not have any assigned task, $\mathbf{x}_{\text{start}}^i$ is the same as the robot's current resource vector. Otherwise, $\mathbf{x}_{\text{start}}^i$ is set to the predicted resource vector after the robot completes its last assigned task. The residual resource prediction function returns the predicted resource vector after visiting the refill stations in $\mathbf{s}_j^i$ and completing task j, given the initial resource vector as $\mathbf{x}_{\text{start}}^i$.

Similar to the prediction step of the extended Kalman filter, the RCG algorithm utilizes a (first order) Taylor expansion to estimate the covariance of the predicted resource vector [34]. From (4), the mean and the covariance of the predicted resource vector, i.e., $\boldsymbol{\mu}_j^i$ and Σ_j^i, respectively, are calculated as

$$\begin{aligned} \boldsymbol{\mu}_j^i &= g\left(\boldsymbol{\mu}_{\text{start}}^i, \mathbf{s}_j^i\right) \\ \Sigma_j^i &= G_j \Sigma_{\text{start}}^i G_j^T + R_j^i \end{aligned} \tag{5}$$

with

$$G_j = \frac{\partial g\left(\boldsymbol{\mu}_{\text{start}}^i, \mathbf{s}_j^i\right)}{\partial \boldsymbol{\mu}_{\text{start}}^i} \tag{6}$$

where $\boldsymbol{\mu}_{\text{start}}^i$ and Σ_{start}^i are the mean and the covariance of the resource vector at the start location, respectively; and G_j is the Jacobian of $g(\cdot)$ evaluated at $\boldsymbol{\mu}_{\text{start}}^i$. Using the predicted resource vector, the competence probability of the lth resource for task j, i.e., $p_{j,l}^i$, is defined as

$$p_{j,l}^i = P\left(x_{j,l}^i > \gamma_l^i\right) = \frac{1}{2}\left[1 - \text{erf}\left(\frac{\gamma_l^i - \mu_{j,l}^i}{\sqrt{2}\sigma_{j,l}^i}\right)\right] \tag{7}$$

with

$$\text{erf}(x) = \frac{1}{\sqrt{\pi}} \int_{-x}^{x} \exp(-t^2)\, dt \tag{8}$$

where $\mu_{j,l}^i$ and $\sigma_{j,l}^i$ are the mean and the standard deviation of the lth predicted resource level, respectively; and γ_l^i is the threshold of the lth resource level. If $\mu_{j,l}^i$ is equal to γ_l^i, the competence probability is 0.5 regardless of the standard deviation of the predicted resource level. However, if $\mu_{j,l}^i$ is greater than γ_l^i, the competence probability decreases as the standard deviation of the predicted resource level increases. This represents that the competence probability takes into account the measurement and prediction uncertainties of the resource level. Note that P_T should be higher than 0.5 to keep the predicted resource level greater than its threshold level.

As shown in Fig. 2, if any of the competence probabilities of the resources is the same or lower than P_T, the refill station for the resource is added to the station list. In order to plan the optimal visiting order of the stations on the list, the depth-first branch-and-bound (DFBB) search algorithm is applied [35], [36]. The idea of the DFBB search is to make the depth-first search more efficient by keeping track of the lowest cost solution found so far. The root node in the search tree represents the start location of the robot, and the node at the nth level represents the station that the robot plans to visit in the nth order. The cost of the nth level, i.e., $n \in \{2, \dots, |\mathbf{s}_j^i|\}$, represents the sum of the costs from the start location to the station at the nth level in the search tree, where $|\mathbf{s}_j^i|$ denotes the cardinality of the station list. The cost of the nth level of robot i for task j, i.e., $c_{j,n}^i$, is defined as

$$c_{j,n}^i = c^i\left(L_{\text{start}}, s_{j,1}^i\right) + c_{1:n}^i, \qquad n \in \left\{2, \dots, \left|\mathbf{s}_j^i\right|\right\} \tag{9}$$

with

$$c_{1:n}^i = \sum_{q=1}^{n-1} c^i\left(s_{j,q}^i, s_{j,q+1}^i\right) \tag{10}$$

where L_{start} is the start location of task j, $s_{j,q}^i$ is the qth station in $\mathbf{s}_j^i$, and $c^i(a, b)$ is a nonnegative cost function that returns the estimated cost for robot i to move from location a to b. Since the node of the last level of the tree is task j, the cost of robot i for task j while visiting the stations in $\mathbf{s}_j^i$ is defined as

$$c_j^i = \begin{cases} c^i(L_{\text{start}}, j), & \text{if } \mathbf{s}_j^i = \emptyset \\ c^i\left(L_{\text{start}}, s_{j,1}^i\right) + c^i\left(s_{j,1}^i, j\right), & \text{if } \left|\mathbf{s}_j^i\right| = 1 \\ c^i\left(L_{\text{start}}, s_{j,1}^i\right) + c_{1:\left|\mathbf{s}_j^i\right|}^i + c^i\left(\mathbf{s}_{j,\left|\mathbf{s}_j^i\right|}^i, j\right), & \text{otherwise} \end{cases} \tag{11}$$

where $c^i(L_{\text{start}}, j)$ is the cost of robot i for task j by taking the direct path, i.e., without visiting any stations. Since the DFBB algorithm keeps the lowest cost solution found so far, the costs of the partial branches in the search tree are compared with the lowest cost, and whenever the cost of the branch is equal to or exceeds the lowest cost, the branch is pruned from the search tree. Using the DFBB algorithm, the optimal sequence of visiting stations can be found. If all of the competence probabilities of the resources are higher than P_T, the sequence of the branch with the lowest cost becomes the visiting order of the stations, and the cost of the branch is regarded as the final cost for the task.

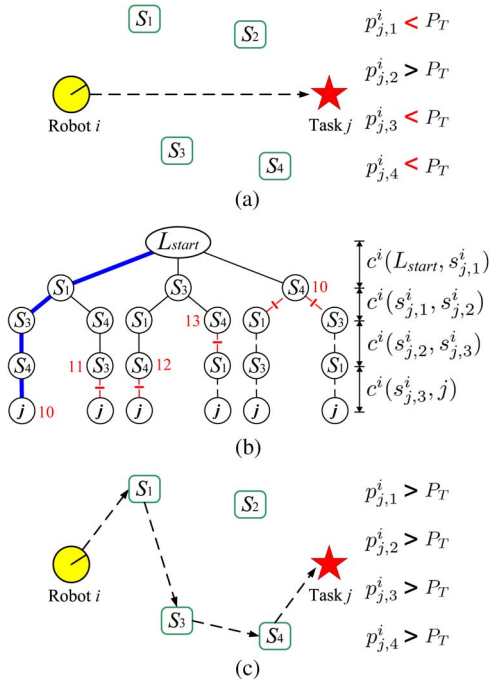


Fig. 3. Illustrative example of the RCG algorithm. (a) Robot i calculates the competence probabilities of its resources when it takes the direct path to task j and realizes that the competence probabilities of resources 1, 3, and 4 are lower than P_T. (b) Optimal visiting sequence of the three stations, namely, S_1, S_3, and S_4, to task j is found by using DFBB, where the costs at the right side of the search tree represent the costs of the branches. (c) Robot calculates the competence probabilities by taking the optimal visiting sequence from (b).

B. Illustrative Example

An example of the RCG algorithm is illustrated in Fig. 3, where robot i consumes four resources to perform task j, and the four refill stations are located apart from each other. In the figure, S_l is the refill station for the lth resource, $\star$ is the location of task j, and the dotted arrows represent the path that the robot plans. First, the robot calculates the competence probabilities of its resources without visiting any stations, as shown in Fig. 3(a), and detects that the competence probabilities of the first, third, and fourth resources are lower than P_T. The robot then determines the optimal sequence of visiting the stations by using DFBB, as shown in Fig. 3(b). The numbers next to the nodes in the figure represent the costs of the paths from the start location to the nodes, and the branches that have the same or higher costs than the minimum cost, 10 in this example, are pruned from the search tree. As a result, the visiting order of the stations with minimum cost is $L_{\text{start}} \rightarrow S_1 \rightarrow S_3 \rightarrow S_4 \rightarrow$ task j, as shown in Fig. 3(b). Since all the competence probabilities are higher than P_T by taking the optimal visiting order, the cost of the task with the path in Fig. 3(c) becomes the final cost for the task.

IV. ANTA

A. Algorithm Description

In the ANTA algorithm, any robot that wants to allocate a task becomes a root robot and broadcasts a trade message for the task. The robots within a certain number of communication hops away from the root robot then construct the minimal spanning tree using the broadcast tree communication with limited tree size. While constructing the tree network, each robot exchanges its winner information, each of which consists of the ID and the cost of the robot that has the lowest cost, with its parent and children, through multihop ad hoc communication. After waiting a certain amount of time, the ID of the winner in the winner information converges to a single robot, and the task is assigned to the winner.

Since the network topology with mobile robots varies with time, it is necessary to limit the size of the network tree. Otherwise, it will expand until it reaches the leaf node such that it requires more time for the winner information for the task to be converged and the network is more likely to be disconnected during the task allocation process. In order to limit the size of the network tree, each trade message contains the message broadcaster's level and the tree level of the task. The trade message for task j, i.e., M_j, is defined as

$$M_j = \begin{bmatrix} j & i_j^\star & c_j^{i_j^\star} & L & L_j \end{bmatrix} \tag{12}$$

where j is the task ID; $i_j^\star$ and $c_j^{i_j^\star}$ are the ID and the cost of the winner for task j, respectively; L is the level of the message broadcaster, and L_j is the tree level of task j. In the message, L and L_j are used to limit the size of the network tree. If a robot receives a trade message that has a smaller L than L_j, it sets its level as $L+1$ and participates in the task allocation process. On the other hand, if L is the same as L_j, the robot is not allowed to participate in the task allocation process. However, if there is no suitable robot that is capable of performing the task, the root robot increases the tree level by one at a time and repeats the task allocation process to improve the chances of finding a capable robot in the extended network tree. Note that the tree level is allowed to be increased up to the predefined maximum tree level.

The robots in the network tree exchange trade messages until the winner information for the task is converged. The winner determination time is defined as the convergence time of the winner information such that each robot in the network tree waits until the winner determination time and then decides the winner for the task. Since the robots receive the first trade message at different times due to the different hop distances from the root robot, each has its own winner determination time such that the winner determination time of robot i for task j is defined as

$$t_j^i = t_c^i + L_j t_I \tag{13}$$

with

$$t_I = 2t_H + t_P \tag{14}$$

where t_c^i is the time when robot i first receives the trade message, t_I is the time interval for exchanging the message between neighbors, t_H is the communication latency per hop, and t_P is the robot process time.

Fig. 4 shows a flowchart of the ANTA algorithm for a robot. When a robot receives a trade message for a new task, it starts the task allocation process only when the message broadcaster's level is smaller than the tree level. The robot sets its parent to the

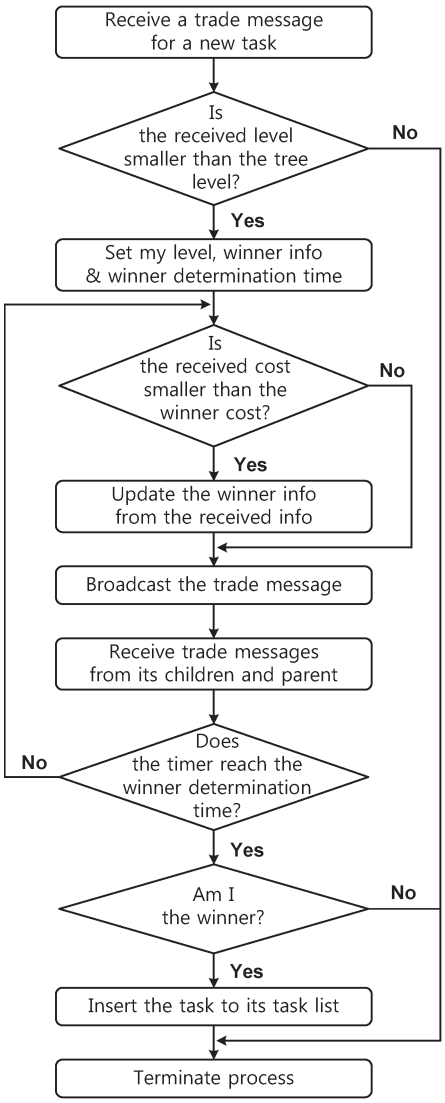


Fig. 4. Flowchart of the ANTA algorithm for a robot.

message broadcaster, assigns its own level to be the level of the received message plus one, and sets the winner determination time from (13). It also sets the winner information to its own ID and cost. While waiting until the winner determination time, the robot exchanges the trade message with its parent and children and updates the winner information whenever the cost from the received message is lower than the cost in the winner information. At the winner determination time, the robot checks whether the ID in the winner information is its own and terminates the process. If it is the winner, it adds the task to its task list.

B. Illustrative Example

A simple example of the ANTA algorithm is shown in Fig. 5. In this example, the costs of robots 1, 2, 3, 4, 5, and 6 are 10, 8, 7, 3, 8, and 4, respectively. The root robot is robot 1, and the tree level is set to 2. In Fig. 5(a), robot 1 first broadcasts its trade message to its neighbors. The pair (ID, cost) next to each node in the figure represents the ID and the cost of the winner robot's information. In Fig. 5(b), since both robots 2 and 3 have

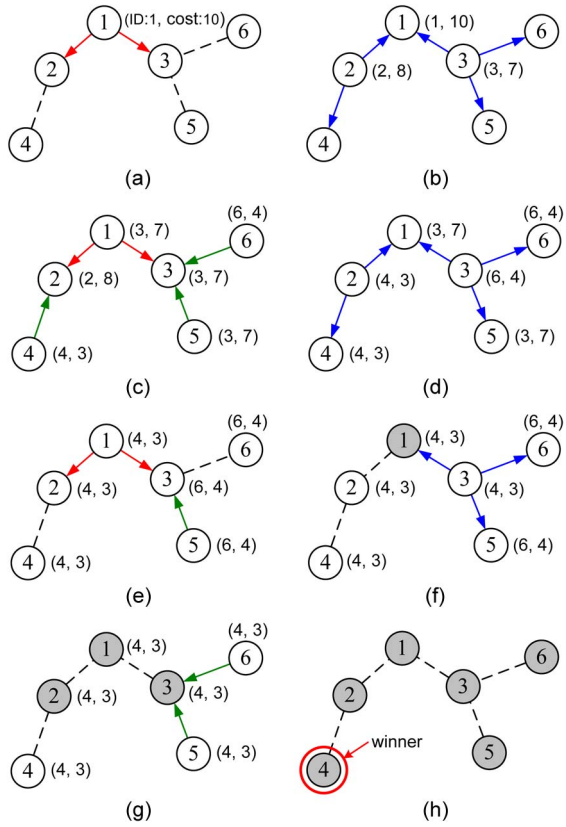
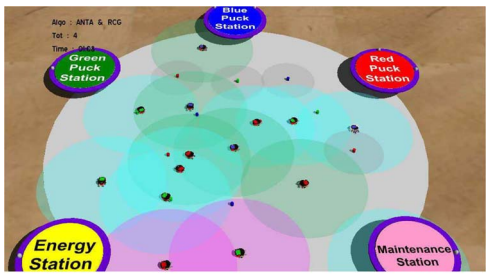
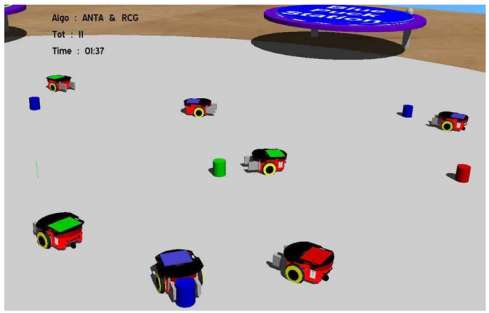


Fig. 5. Illustrative example of the ANTA algorithm. The white nodes represent the robots that are processing the ANTA algorithm, and the gray nodes represent the robots that completed the task allocation. The dotted lines are the network connections among the robots, and the arrows show the direction of messages. The numbers in the nodes are the robot IDs, and the pair next to each node is the ID and the cost of the winner information of the node. For example, (1,10) in (a) represents that robot 1 currently regards itself as the winner, and its cost is 10.

lower costs than robot 1, each considers itself as the winner, and thus, robots 2 and 3 broadcast (2,8) and (3,7), respectively. Since robot 3 has lower cost than robots 1 and 2, robot 1 updates its pair from (1,10) to (3,7) after receiving the messages from robots 2 and 3, as shown in Fig. 5(c). In the case of robot 4, since its cost is lower than the received cost from robot 2, it sets its pair to (4,3) and broadcasts it, as shown in Fig. 5(c). Likewise, robot 6 also sets its pair to its ID and cost, i.e., (6,4). In the case of robot 5, however, its cost is higher than the received cost, and therefore, it sets its pair to the received pair, i.e., (3,7), as shown in Fig. 5(c). In Fig. 5(d), robots 2 and 3 update their pairs to (4,3) and (6,4), respectively, and broadcast their messages. In Fig. 5(e), the pairs of the robots in the left branch, i.e., robots 2 and 4, are converged to (4,3), and the pairs of the robots in the right branch, i.e., robots 3, 5, and 6, are converged to (6,4). The root robot then selects the pair with lower cost, i.e., (4,3), and broadcasts its message to its neighbors. In Fig. 5(f), robot 1 terminates its task allocation upon realizing that robot 4 is the winner. After receiving the message from robot 1, robot 3 also updates its pair to (4,3) and broadcasts its message. In Fig. 5(g), the pairs of all the robots are converged to (4,3), and robots 2 and 3 terminate their task allocation processes. In Fig. 5(h), the rest of the robots also terminate the task allocation processes, and robot 4 becomes the winner for the task.



(a)



(b)

Fig. 6. Screenshot of the multirobot foraging simulation. (a) Five refill stations, i.e., energy station; maintenance station; and red, green, and blue puck stations, are located around the arena. (b) Heterogeneous robots collect pucks with different colors and weights. The circle around each robot represents the communication range.

V. Multirobot Foraging

The foraging task can be considered as a simplified version of complex applications, such as autonomous exploration and distributed surveillance. In a classical foraging task, a group of robots collect preys from an environment and either consume or return them to the nest. The robots regulate their behaviors, either foraging or resting, and they receive energy from the collected preys. The task allocation in multirobot foraging is mainly obtained through the use of finite state machines with different state transition strategies. Examples of these strategies include a bioinspired approach, a threshold-based mechanism, a behavior-based approach, and a probabilistic approach [37]–[40]. In order to demonstrate the effectiveness of the proposed approach, a modified multirobot foraging scenario is considered in this paper. The modified version differs from the traditional one in two aspects. First, the robots do not get energy from their collected preys. Instead, they can refill their resources from refill stations. Second, the robots forage most of the time, and they only rest for a short time in the refill stations while refilling their resources.

The simulations were carried out using a commercially available robot simulator called Webots [41]. Fig. 6 shows a screenshot of the foraging simulation, where 15 heterogeneous robots collect pucks with different colors and weights. The pucks continuously appear in an arena with a 10.0-m radius at any time. The locations of the pucks are not known *a priori* and must be discovered by the robots in real time. The goal of the foraging mission is to allocate the puck collecting tasks to the robots that can minimize the task completion time. During the simulation, the robots randomly move around the arena searching for the pucks. If a robot senses a puck, it informs this to its neighbors within its communication range by broadcasting the puck collection task message. The cost of the robot for the task is defined as the estimated task completion time. While all robots are equally capable of sensing any type of puck (regardless of their colors and weights), each robot can collect only a specific type of puck.

A. Robot Resources

The robots use three types of resources, namely, energy, expendable components, and live load. The robots consume energy whenever they move around and pick up the pucks. When the energy level falls below a threshold, the robot should stop foraging and visit the energy station to recharge its energy. The robot also must replace its expendable components in the maintenance station once in every predefined period. When a robot loads up a puck, the live load of the robot increases. If the live load exceeds the predefined allowable load, the robot should empty its pucks in the puck station with the same color.

B. Algorithms

The overall performance results of five task allocation algorithms, namely, sequential single-item auctions (SSIA) [42], repeated sequential single-item auctions (RSSIA) [43], MURDOCH [23], ANTA$_1$ with RCG, and ANTA$_4$ with RCG, were compared and analyzed. The subscript in ANTA represents the maximum tree level. The details of the algorithms are described in the following.

- *SSIA*: In SSIA, the robot that has found a puck becomes an auctioneer and broadcasts an auction message to its direct neighbors, i.e., the robots that are located in a single hop away from the auctioneer. Any capable robots that have received the message become bidders and generate bids for the task considering the sequential execution order of their previously assigned tasks. After receiving the bids from the bidders, the auctioneer selects the winner, which has the lowest cost, and allocates the task to the winner by broadcasting the assignment message.
- *RSSIA*: The RSSIA is a modified version of the SSIA such that the robot reauctions the rest of its assigned tasks whenever it completes a task.
- *MURDOCH*: The MURDOCH uses a sequence of first-price one-round auctions. Unlike the SSIA and RSSIA, in which the robots do not consider their resources in task allocation, the robot with MURDOCH will not participate in the auction process when its resource levels are sufficiently low. Instead, it visits refill stations to refill the depleted resources.
- ANTA$_1$ *with RCG and* ANTA$_4$ *with RCG*: In order to compare the proposed approach with different maximum tree levels, the two algorithms are compared. In the case when the maximum tree level is one, only the robots that are located a single hop away from the root robot are allowed to participate in the task allocation process. In the case of the maximum tree level with four, the network tree can be extended up to four levels until finding the capable robot in the network tree. Note that P_T for the RCG algorithm was set to 0.6.

TABLE I
TASK TYPES AND CORRESPONDING CAPABLE ROBOTS

Robot *ID*	1,2,3	4,5	6,7,8	9,10	11,12,13	14,15
Task type	R&L	R&H	G&L	G&H	B&L	B&H

C. Simulation Results

1) Results With Different CCRs: In the first simulation experiment, the results of the algorithms were compared by changing the communication coverage rate (CCR), which is defined as

$$CCR = \frac{\text{Area of communication coverage}}{\text{Area of arena}} \cdot 100\,(\%) \quad (15)$$

where the area of the arena is 100π m^2 in the simulation, and the area of a communication coverage is πr^2 m^2 with the communication radius r m. The CCR is related with the probability of finding a capable robot in a single-hop communication range. The task type is defined by the color and weight of the puck, and each puck has different color types, namely, red (R), green (G), and blue (B), and different weight types, namely, light (L) and heavy (H). Table I shows the task types and the corresponding capable robots. For example, task type R&L represents the red-colored lightweight puck collecting task, and only the robots with IDs from 1 to 3 are capable of performing this task. Since the number of neighbors of each robot becomes larger as the CCR increases, the probability of finding a capable robot in a single-hop communication range is also increased. On the other hand, the chance of finding a capable robot becomes lower as the CCR decreases.

In order to examine the effect of the CCR in task allocation, the communication ranges were set to 3.2, 4.5, 5.5, 6.3, and 7.1 m for CCRs with 10%, 20%, 30%, 40%, and 50%, respectively. The sensing range of the robots was set to 1.6 m, which is half of the minimum communication range. The simulation was terminated when the 100th task was completed, and 30 simulation runs were performed with each CCR. Fig. 7(a) shows the average task completion time of the algorithms with respect to five different CCRs. Since the robots with the RCG algorithm were capable of planning whether to visit the refill stations before starting the tasks, they were able to minimize extra time delay due to resource depletion during task execution. As a result, the overall task completion time of the algorithms with RCG, i.e., ANTA$_1$ with RCG and ANTA$_4$ with RCG, was lower than that of the other algorithms. Moreover, since the ANTA$_4$ algorithm is capable of extending the network tree up to four tree levels, it had higher probability of finding the capable robot in the network than SSIA, RSSIA, MURDOCH, and ANTA$_1$. However, as the CCR was increased to 50%, the difference between the results of ANTA$_1$ and ANTA$_4$ decreased. This can be explained by the probability that the capable robot is within a single-hop communication range. The single-hop winner probability, i.e., P_S, is defined as

$$P_S = 1 - \left(\frac{1}{m}\right)^n \quad (16)$$

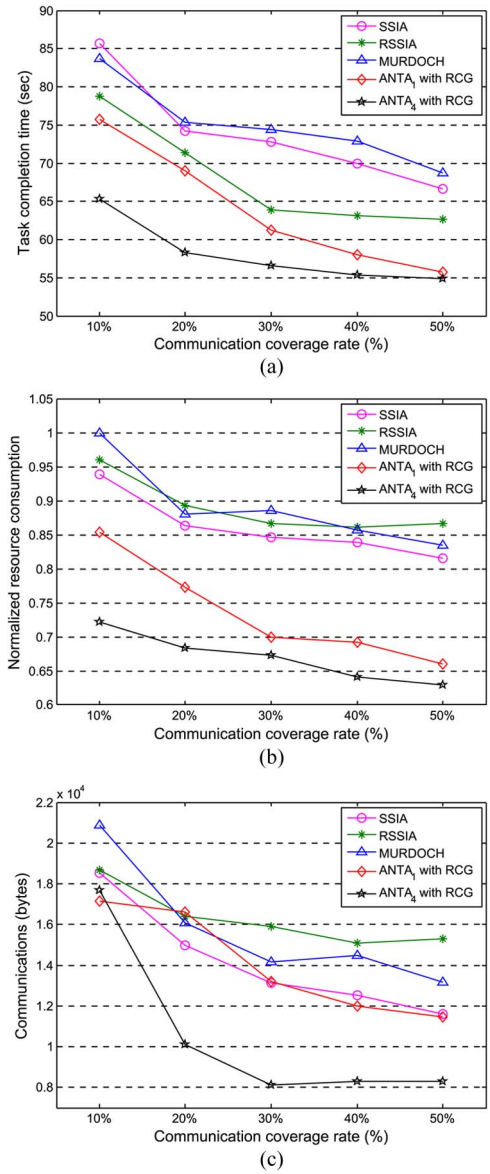


Fig. 7. Simulation results with different CCRs. (a) Average task completion time. (b) Average normalized resource consumption. (c) Average amount of communications.

where n is the number of direct neighbors, i.e., the number of robots within a single-hop communication range, and m is the number of task types. It is assumed that the probability that a robot is capable of performing one type of task out of m types is $1/m$. For the first simulation, the number of task types m is six, as shown in Table I. Since n is proportional to the CCR, P_S decreases with a decrease in the CCR. In the case of ANTA$_1$, only the direct neighbors were allowed to participate in task allocation, and consequently, the algorithm suffered from a misassignment problem when the CCR was low. However, as the CCR was increased to 50%, P_S also increased, i.e., the likelihood of finding a capable robot in a single hop was improved. Thus, the results of ANTA$_1$ and ANTA$_4$ were almost the same when the CCR was 50%. Likewise, the results of the other algorithms also decreased with an increase in the CCR, as shown in Fig. 7(a).

In order to compare the amount of resource consumption between the algorithms, the normalized resource consumption

of algorithm a, i.e., $\bar{E}^a$, is defined as

$$\bar{E}^a = \frac{1}{N_L} \sum_{l \in \mathcal{L}} \frac{E_l^a}{\max_{a'} E_l^{a'}} \tag{17}$$

with

$$E_l^a = \sum_{r \in \mathcal{R}} E_{i,l}^a,$$

$$a \in \{\text{SSIA},\ \text{RSSIA},\ \text{MURDOCH},\ \text{ANTA}_1 \text{ with RCG}$$

$$\text{ANTA}_4 \text{ with RCG}\} \tag{18}$$

where $E_{i,l}^a$ is the total consumption of the lth resource of robot i with algorithm a. Fig. 7(b) shows the normalized resource consumption of the algorithms for different CCRs. Since the robots with RCG were less prone to resource depletion in the middle of task execution, they had less instances of extra resource consumption than the other algorithms. Similar to the results in Fig. 7(a), the ANTA_4 algorithm consumed less resources than the ANTA_1 algorithm because it could find the capable robots faster than ANTA_1.

Fig. 7(c) shows the average amount of communications of the algorithms with different CCRs. In the simulation, the robot that has sensed a puck broadcasts messages until either the task is allocated or the puck is out of the robot's sensing range. Since the ANTA_4 algorithm completed the tasks faster than the others, the amount of communications of the algorithm was lower than those of the other algorithms. The results of the first simulation demonstrate that the combination of ANTA_4 and RCG can allocate tasks more efficiently with less task completion time, lower resource consumption, and fewer communications than all the other algorithms through all values of the CCRs.

2) Results With Different Numbers of Task Types: In the second simulation experiment, the effect of different numbers of task types on the five algorithms was tested and analyzed with the fixed CCR of 30%. The numbers of task types were set to 3, 6, 9, 12, and 15; and the corresponding team of heterogeneous robots was also divided into 3, 6, 9, 12, and 15, groups, respectively. The robots in each group are capable of one type of task only. For example, in the case where the number of task types is 3, the robots are divided into three groups, and the robots in group 1, group 2, and group 3 can collect only type-1, type-2, and type-3 pucks, respectively.

Fig. 8(a) shows the average task completion time of the algorithms for different numbers of task types. The average task completion time increased as the number of task types increased. This was because it took more time to find a capable robot among the heterogeneous robots as the number of task types increased. Similar to the first simulation results, the graph shows that ANTA_4 with RCG resulted in the lowest task completion time. Since increasing the number of task types causes lower P_S in (16), the single-hop algorithms, i.e., SSIA, RSSIA, MURDOCH, and ANTA_1 with RCG, resulted in longer task completion time than ANTA_4 with RCG. The normalized resource consumption in Fig. 8(b) and the amount of communications in Fig. 8(c) also show similar pattern as in Fig. 8(a). As a result, ANTA_4 with RCG outperforms the other algorithms in terms of task completion time, resource consumption, and communication load.

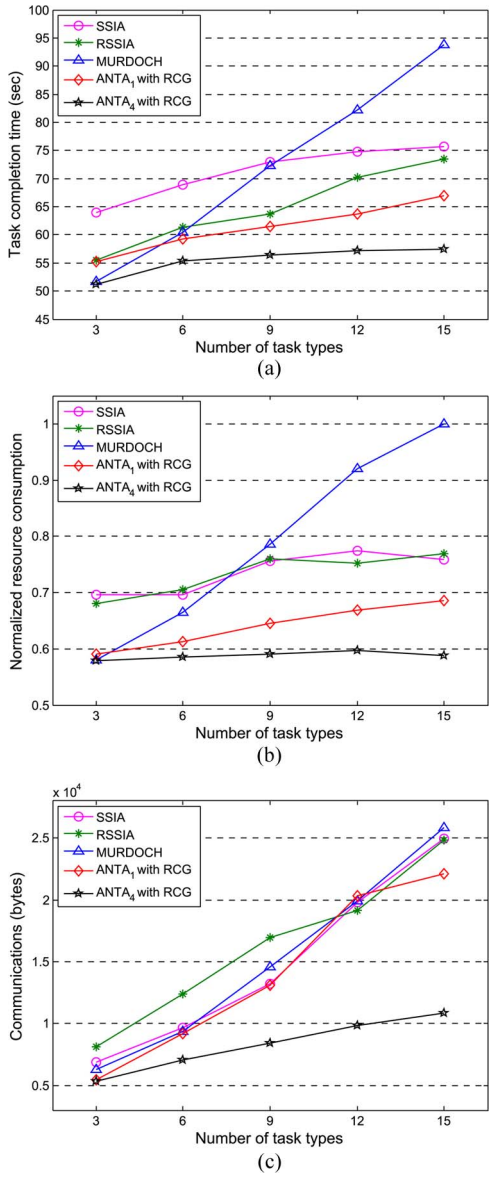


Fig. 8. Simulation results with different numbers of task types. (a) Average task completion time. (b) Average normalized resource consumption. (c) Average amount of communications.

3) Results With Different CCRs and Numbers of Task Types: In the third simulation experiment, the task completion times of the algorithms with different combinations of the CCR and the number of task types were compared. Fig. 9 shows the simulation results of SSIA, RSSIA, MURDOCH, ANTA_1 with RCG, and ANTA_4 with RCG, respectively. As shown in the figures, the overall task completion time was increased either by a decrease in the CCR or by an increase in the number of task types. The graphs also show that the algorithms without using RCG had higher average task completion time and had steeper slope than those with RCG. In order to numerically show the difference between the algorithms, the average task completion time and the average slope of the graphs are presented in Table II. The table also shows that ANTA_4 with RCG is more consistent than the other algorithms with different CCRs and numbers of task types.

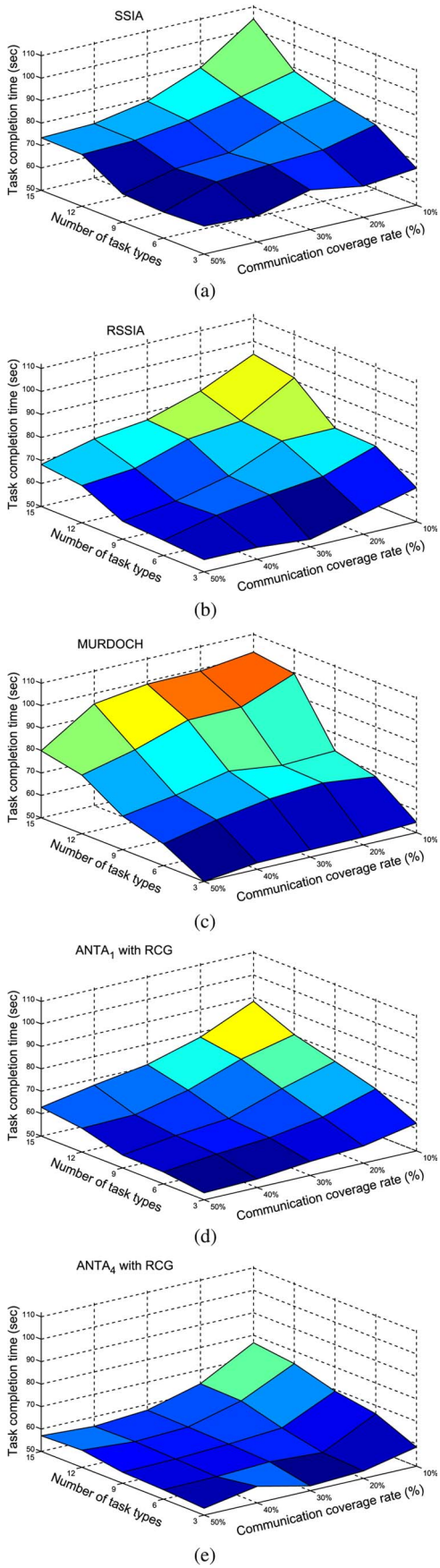


Fig. 9. Average task completion time with different CCRs and different numbers of task types. (a) SSIA. (b) RSSIA. (c) MURDOCH. (d) $ANTA_1$ with RCG. (e) $ANTA_4$ with RCG.

TABLE II
AVERAGE TIME AND AVERAGE SLOPE OF TASK COMPLETION TIME

Algorithm	Average time (sec)	Average slope
SSIA	74.1	1.8
RSSIA	69.2	1.7
MURDOCH	74.4	2.2
$ANTA_1$ with RCG	64.9	1.5
$ANTA_4$ with RCG	58.9	1.0

VI. CONCLUSION

This paper has proposed the RCG and ANTA algorithms for MRTA. The RCG algorithm enabled a robot to keep its resource levels higher than the thresholds after completing a task. Using this algorithm, unexpected increase in task completion time, resource consumption, and communication load due to depletion of resources during task execution could be minimized. The ANTA algorithm constructed a minimal spanning tree network and found a capable robot in the tree that has the lowest cost for the task. Since this algorithm uses a multihop ad hoc communication network in task allocation, it is more efficient in finding a capable robot with a limited communication range than the other algorithms that use single-hop communication. The proposed approach was tested in multirobot foraging simulations with different CCRs and numbers of task types. The simulation results demonstrated the effectiveness of the proposed approach in terms of task completion time, resource consumption, and communication load.

For future work, a method to select the maximum task tree level in ANTA should be developed for efficient and robust task allocation. In addition, heuristic algorithms should be investigated to improve the DFBB search in the RCG algorithm for a large number of resources. Finally, the proposed approach should be implemented on physical robots and tested in a real environment to demonstrate its effectiveness and robustness.

REFERENCES

[1] L. E. Parker, "Distributed intelligence: Overview of the field and its application in multi-robot systems," *J. Phys. Agent*, vol. 2, pp. 5–14, 2008.
[2] J. C. Elizondo-Leal, G. Ramirez-Torres, and G. T. Pulido, "Multi-robot exploration and mapping using self biddings," in *Proc. Adv. Artif. Intell.*, 2008, pp. 392–401.
[3] W. Xu, R. Jiang, and Y. Chen, "Map alignment based on PLICP algorithm for multi-robot SLAM," in *Proc. IEEE Symp. Ind. Electron.*, 2012, pp. 926–930.
[4] I. Bayezit and B. Fidan, "Distributed cohesive motion control of flight vehicle formations," *IEEE Trans. Ind. Electron.*, vol. 60, no. 12, pp. 5763–5772, Dec. 2013.
[5] Y. Toda and N. Kubota, "Self-localization based on multiresolution map for remote control of multiple mobile robots," *IEEE Trans. Ind. Informat.*, vol. 9, no. 3, pp. 1772–1781, Aug. 2013.
[6] B. Ranjbar-Sahraei, F. Shabaninia, A. Nemati, and S.-D. Stan, "A novel robust decentralized adaptive fuzzy control for swarm formation of multiagent systems," *IEEE Trans. Ind. Electron.*, vol. 59, no. 8, pp. 3124–3134, Aug. 2012.
[7] L. Doitsidis *et al.*, "Optimal surveillance coverage for teams of micro aerial vehicles in GPS-denied environments using onboard vision," *Auton. Robots*, vol. 33, no. 1/2, pp. 173–188, Aug. 2012.
[8] Z. Wang and D. Gu, "Cooperative target tracking control of multiple robots," *IEEE Trans. Ind. Electron.*, vol. 59, no. 8, pp. 3232–3240, Aug. 2012.

[9] B. P. Gerkey and M. J. Matarić, "A formal analysis and taxonomy of task allocation in multi-robot systems," *Int. J. Robot. Res.*, vol. 23, no. 9, pp. 939–954, Sep. 2004.
[10] F. Cordes *et al.*, "LUNARES: Lunar crater exploration with heterogeneous multi robot systems," *Intell. Serv. Robot.*, vol. 4, no. 1, pp. 61–89, Jan. 2011.
[11] P. R. Wurman, R. D'andrea, and M. Mountz, "Coordinating hundreds of cooperative, autonomous vehicles in warehouses," *Artif. Intell. Mag.*, vol. 29, pp. 9–19, 2008.
[12] K. C. T. Vivaldini *et al.*, "Robotic forklifts for intelligent warehouses: Routing, path planning, and auto-localization," in *Proc. IEEE Conf. Ind. Technol.*, 2010, pp. 1463–1468.
[13] B. Mazzolai *et al.*, "Networked and cooperating robots for urban hygiene: The EU funded DustBot project," in *Proc. Int. Conf. Ubiquitous Robot. Ambient. Intell.*, 2008, pp. 447–452.
[14] J. B. Grau *et al.*, "Sustainable agriculture using an intelligent mechatronic system," in *Proc. IEEE Conf. Ind. Elec.*, 2009, pp. 3416–3421.
[15] G. Song, H. Wang, J. Zhang, and T. Meng, "Automatic docking system for recharging home surveillance robots," *IEEE Trans. Consum. Electron.*, vol. 57, no. 2, pp. 428–435, May 2011.
[16] A. Carlini *et al.*, "Analysis and design in providing a robotised cleaning and validation system for hospital environment," in *Proc. Int. Conf. Adv. Serv. Comput.*, 2012, pp. 58–63.
[17] D. C. Slaughter, D. K. Giles, and D. Downey, "Autonomous robotic weed control systems: A review," *Comput. Electron. Agric.*, vol. 61, no. 1, pp. 63–78, Apr. 2008.
[18] M. K. Lee, S. Kiesler, J. Forlizzi, S. Srinivasa, and P. E. Rybski, "Gracefully mitigating breakdowns in robotic services," in *Proc. IEEE Conf. HRI*, 2010, pp. 203–210.
[19] A. Beynier and A.-I. Mouaddib, "Decentralized Markov decision processes for handling temporal and resource constraints in a multiple robot system," *Distrib. Auton. Robot. Syst.*, vol. 6, pp. 191–200, 2007.
[20] M. Strens and N. Windelinckx, "Combining planning with reinforcement learning for multi-robot task allocation," in *Proc. Adap. Agents Multi-Agent Syst. II*, 2005, pp. 260–274.
[21] D.-H. Lee, J.-H. Han, and J.-H. Kim, "Market-based multiagent framework for balanced task allocation," in *Proc. Adv. Intell. Syst. Comput.*, 2013, vol. 208, pp. 549–559.
[22] B. Kaleci and O. Parlaktuna, "Performance analysis of bid calculation methods in multirobot market-based task allocation," *Turkish J. Elect. Eng. Comput. Sci.*, vol. 21, no. 2, pp. 565–585, 2013.
[23] B. P. Gerkey and M. J. Matarić, "Sold!: Auction methods for multirobot coordination," *IEEE Trans. Robot. Autom.*, vol. 18, no. 5, pp. 758–768, Oct. 2002.
[24] M. B. Dias, R. Zlot, N. Kalra, and A. Stentz, "Market-based multirobot coordination: A survey and analysis," *Proc. IEEE*, vol. 94, no. 7, pp. 1257–1270, Jul. 2006.
[25] R. Zlot and A. Stentz, "Market-based multirobot coordination for complex tasks," *Int. J. Robot. Res.*, vol. 25, no. 1, pp. 73–101, Jan. 2006.
[26] G. Li, Y. Tamura, M. Wu, A. Yamashita, and H. Asama, "Hybrid dynamic mobile task allocation and reallocation methodology for distributed multirobot coordination," in *Proc. IEEE/ASME Conf. Adv. Intell. Mechatronics*, 2012, pp. 190–195.
[27] M. A. Batalin and G. S. Sukhatme, "Sensor network-based multirobot task allocation," in *Proc. IEEE Conf. Intell. Robot. Syst.*, 2003, pp. 1939–1944.
[28] M. A. Batalin and G. S. Sukhatme, "Using a sensor network for distributed multi-robot task allocation," in *Proc. IEEE Conf. Robot. Autom.*, 2004, pp. 158–164.
[29] J. McLurkin, "Analysis and implementation of distributed algorithms for multi-robot systems," M.S. thesis, Massachusetts Inst. Technol., Cambridge, MA, USA, 2008.
[30] J. McLurkin and D. Yamins, "Dynamic task assignment in robot swarms," *Proc. Robot. Sci. Syst.*, 2005.
[31] D. Li, X. Jia, and H. Liu, "Energy efficient broadcast routing in static ad hoc wireless networks," *IEEE Trans. Mobile Comput.*, vol. 3, no. 2, pp. 144–151, Apr.–Jun. 2004.
[32] A. Jüttner and Á. Magi, "Tree based broadcast in ad hoc networks," *Mobile Netw. Appl.*, vol. 10, no. 5, pp. 753–762, Oct. 2005.
[33] I. Maric and R. D. Yates, "Cooperative multihop broadcast for wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 22, no. 6, pp. 1080–1088, Aug. 2004.
[34] G. A. Einicke and L. B. White, "Robust extended Kalman filtering," *IEEE Trans. Signal Process.*, vol. 47, no. 9, pp. 2596–2599, Sep. 1999.
[35] K. S. Macarthur, R. Stranders, S. D. Ramchurn, and N. R. Jennings, "A distributed anytime algorithm for dynamic task allocation in multi-agent systems," in *Proc. AAAI Conf. Artif. Intell.*, 2011, pp. 701–706.
[36] B. Cai *et al.*, "Multiobjective optimization for autonomous straddle carrier scheduling at automated container terminals," *IEEE Trans. Autom. Sci. Eng.*, vol. 10, no. 3, pp. 711–725, Jul. 2013.
[37] T. H. Labella, M. Dorigo, and J.-L. Deneubourg, "Division of labor in a group of robots inspired by ants' foraging behavior," *ACM Trans. Auton. Adap. Syst.*, vol. 1, no. 1, pp. 4–25, Sep. 2006.
[38] M. J. B. Krieger and J.-B. Billeter, "The call of duty: Self-organised task allocation in a population of up to twelve mobile robots," *Robot. Auton. Syst.*, vol. 30, pp. 65–84, 2000.
[39] K. Lerman, C. Jones, A. Galstyan, and M. J. Matarić, "Analysis of dynamic task allocation in multi-robot systems," *Int. J. Robot. Res.*, vol. 25, no. 3, pp. 225–241, Mar. 2006.
[40] W. Liu and A. F. T. Winfield, "Modeling and optimization of adaptive foraging in swarm robotic systems," *Int. J. Robot. Res.*, vol. 29, no. 14, pp. 1743–1760, Dec. 2010.
[41] O. Michel, "Webots: Professional mobile robot simulation," *Int. J. Adv. Robot. Syst.*, vol. 1, no. 1, pp. 39–42, 2004.
[42] S. Koenig *et al.*, "The power of sequential single-item auctions for agent coordination," in *Proc. Nat. Conf. Artif. Intell.*, 2005, pp. 1625–1629.
[43] M. Nanjanath and M. Gini, "Repeated auctions for robust task execution by a robot team," *Robot. Auton. Syst.*, vol. 58, no. 7, pp. 900–909, Jul. 2010.

Dong-Hyun Lee received the B.S. degree in electrical engineering in 2007 from Kyungpook National University, Daegu, Korea, and the M.S. degree in electrical engineering in 2009 from the Korea Advanced Institute of Science and Technology, Daejeon, Korea, where he is currently working toward the Ph.D. degree.

Sheir Afgen Zaheer received the B.S. degree in mechatronics engineering in 2008 from the National University of Sciences and Technology, Islamabad, Pakistan, and the M.S. degree in electrical engineering in 2012 from the Korea Advanced Institute of Science and Technology, Daejeon, Korea, where he is currently working toward the Ph.D. degree.

Jong-Hwan Kim (F'09) received the B.S., M.S., and Ph.D. degrees from Seoul National University, Seoul, Korea, in 1981, 1983, and 1987, respectively, all in electronics engineering.

Since 1988, he has been with the Department of Electrical Engineering, Korea Advanced Institute of Science and Technology, Daejeon, Korea, where he is currently a KT Chair Professor and the Director of the National Robotics Research Center for Robot Intelligence Technology. His research interests include intelligence technology, intelligence super agents, ubiquitous and genetic robots, and humanoid robots.

Dr. Kim is the Founder and currently the President of the Federation of International Robot-soccer Association (FIRA, www.FIRA.net) and the International Robot Olympiad Committee (IROC, www.IROC.org). He is an Associate Editor of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION and the *IEEE Computational Intelligence Magazine*. His name was included in The Barons 500: Leaders for the New Century in 2000 as the Father of Robot Football.