Robotica (2013) volume 31, pp. 923–934. © Cambridge University Press 2013
doi:10.1017/S0263574713000192

# Game theory-based negotiation for multiple robots task allocation

Rongxin Cui†*, Ji Guo‡ and Bo Gao†

†*College of Marine Engineering, Northwestern Polytechnical University, Xi'an 710072, P. R. China*
‡*College of Physics and Electrical Engineering, Anyang Normal University, Anyang 455000, P. R. China*

(Accepted February 5, 2013. First published online: March 7, 2013)

**SUMMARY**
This paper investigates task allocation for multiple robots by applying the game theory-based negotiation approach. Based on the initial task allocation using a contract net-based approach, a new method to select the negotiation robots and construct the negotiation set is proposed by employing the utility functions. A negotiation mechanism suitable for the decentralized task allocation is also presented. Then, a game theory-based negotiation strategy is proposed to achieve the Pareto-optimal solution for the task reallocation. Extensive simulation results are provided to show that the task allocation solutions after the negotiation are better than the initial contract net-based allocation. In addition, experimental results are further presented to show the effectiveness of the approach presented.

KEYWORDS: Task allocation; Game theory; Multiple robots; Negotiation; Cooperative control; Pareto-optimization.

## 1. Introduction

The cooperation of multiple autonomous robots, including unmanned aerial vehicles (UAVs), automatic ground vehicles (AGVs), unmanned underwater vehicles (UUVs), etc., is being employed in a growing number of applications, such as military applications, space/subsea explorations, and disaster relief.[1–7] Through the exchange and sharing of local information, the cooperation of multiple robots can offer improved performances over single robots, such as efficiency, robustness, flexibility and fault tolerance.[1,3,8] Hence, the research on multi-robot systems has been extremely active in recent years, which covers distributed decision making, formation control, area coverage and their applications. When dispatching multiple robots to execute the missions, the first step is to assign individual robots subtasks of a given system-level task, which is called task allocation. The main goal of task allocation is to maximize the overall performance of the system and to fulfill the tasks as soon as possible.[3,9] A typical scenario is that a group of robots is deployed to visit a set of locations/targets for some purpose with routes that minimize the completion time or the distance traveled.[10]

Depending on the nature of task availability, the multiple robots coordination problem can be categorized as static and dynamic.[10] When the tasks to be performed are known to robots before task execution, they can be referred to as static task allocation. For dynamic task allocation, the assignment of robots to subtasks is a dynamic process and may need to be continuously adjusted in response to changes in the task environment or group performance. Numerous schemes have been proposed for the task allocation problems. A formal analysis and taxonomy of multi-robot task allocation are introduced in ref. [11], providing a particular taxonomy for the task allocation problem from several fields, including operations research, economics, scheduling, network flows, and combinatorial optimization. A distributed and asynchronous task allocation for the area coverage of robots is proposed in ref. [12], where each robot is responsible for performing tasks that occur in its Voronoi cell. Inspired by a resource distribution process commonly found in nature, vacancy chain scheduling-based task allocation is proposed in ref. [3]. Coalitions formation for task allocation is studied in ref. [13], where the teams of robots are formed and each team is assigned a particular task to complete the set of tasks in a best way. Clustering-based task allocation is proposed in refs. [10, 14], where the $K$-means clustering is used in ref. [14]. Other clustering algorithms such as the one presented in ref. [15] can also be applied to this problem. Learning is another method for task allocation.[16–18] Assuming that the task stream would be unpredictable, a hybrid planning/learning system combining planning with reinforcement learning that allows scheduling of a group of robots for a heterogeneous stream of tasks is proposed in ref. [16]. A gradient ascent learning algorithm is proposed for task allocation in ref. [17]. An adaptive learning approach is proposed in ref. [18], where the solutions are obtained for coupled Hamilton–Jacobi equations.

Recently, the principles of market economies have been introduced to the coordination of multi-robot systems, including market-based coordination approaches and auction-based task allocation schemes.[19] These approaches have been shown to efficiently produce suboptimal solutions.[20,21] In the market-based mechanisms, the virtual market is constructed based on the tasks in which the tasks are treated as the goods in the virtual market, and the robots are treated as self-interested participants in the virtual market in which they can exchange tasks for payment. Subsequently, the task allocation problem is solved based on the prices in the market. The basic idea of a market-based system is to facilitate task allocation through contract negotiations. A *manager* offers tasks to *contractors,* which may then submit bids based on their abilities to perform the tasks and the highest bid wins the assignment. Each task will only be assigned to a single robot since only one robot is selected by the auctioneer as the winner. A consensus-based auction

* Corresponding author. E-mail: rongxin.cui@gmail.com.

algorithm is proposed for robust task allocation in ref. [21], where the consensus is used to achieve an agreement of the situational awareness for the robots. Auction-based complex task allocation for the mobile surveillance system is proposed in ref. [20], where the priority of each task is considered. The disadvantages of the market-based methodology are that it needs to define the precise price for every task and needs to change the price along the task allocation process. This results in large communication burden and the price needs to be carefully set to achieve the optimal solution.

In order to obtain the Pareto-optimal or Nash equilibrium solution for task allocation problems, a negotiation mechanism can be applied between each pair of robots.[22] The potential benefits of negotiation include its efficiency for computationally intense negotiations searching for optimal results and the ability to incorporate multiple negotiation strategies for the changing environments. Three subproblems need to be solved in the negotiation mechanisms, including determining negotiation robots, rules construction and the negotiation policy.[23] Current multi-agent negotiation is twofold: distributed artificial intelligent and software intelligence design, including the contract-based negotiation,[24,25] planning-based negotiation,[26] market-based negotiation,[27,28] game theory-based negotiation[29,30] and artificial intelligent-based negotiation.[31] In these works, there is a lack of a specific procedure to choose the negotiation robots in the large group of robots and a specific way to construct the negotiation set.

This work focuses on the design of a game theory-based negotiation approach for the task allocation problem. We denote the set of robots and tasks by $\mathcal{A} = \{A_1, \ldots, A_m\}$ and $\mathcal{T} = \{T_1, \ldots, T_n\}$, respectively, and consider only the case where the number of tasks is larger than the number of robots, i.e., $n > m$, where $n$ and $m$ are the number of tasks and robots, respectively. The main motivation behind this consideration is that, for most applications, we would like to use fewer robots to perform tasks. We decompose the task allocation problem into two steps. First, the contract net-based negotiation is employed to obtain the initial task allocation. Second, based on the approach computing the possible task set involving in the further negotiation, game theory-based negotiation is employed to the reallocation of the tasks. This partitioning reduces the dimension of the problem. The main contribution of this work lies in three aspects: (i) we propose a new method to select the negotiation robots and construct the negotiation set by employing the utility functions. This method decreases the computational complexity and is suitable for the case where the computational ability of a single robot is limited; (ii) we propose a negotiation mechanism that is suitable for the decentralized task allocation. This can realize the merit of the decentralized system structure of the multi-robot system; and (iii) a negotiation strategy is proposed to achieve the Pareto-optimal solution for the multi-robot task allocation problem, improving the system without additional cost added for each robot.

The remainder of the paper is organized as follows. In Section 2, problem formulation is presented. The initial task allocation is presented in Section 3, followed by the task reallocation in Section 4. Simulation and experimental results are presented in Sections 5 and 6, respectively. Conclusions are drawn in Section 7.

## 2. Problem Formulation

We consider that all the robots involved in the negotiation are autonomous in the sense that they have their own utility functions, and no global notion of utility plays a role in their design. The robots have disparate goals and are individually motivated. Each robot is rational, i.e., it always maximizes its own utility.[32] The goal of task allocation is, given a set of of tasks $\mathcal{T} = \{T_1, \ldots, T_n\}$ and $m$ robots, $A_1, \ldots, A_m$, to find a conflict-free matching of tasks to robots that minimize some cost. The term conflict-free means that each task is assigned to not more than one robot. The task allocation problem can be written as follows, with binary decision variable $x_{ij}$ that indicates whether task $T_i$ is assigned to agent $A_j$ or not:

$$\min \left\{ \phi_j \left( \sum_{i=1}^{n} w_{ij} x_{ij} \right), \text{ for } j = 1, \ldots, m \right\}, \tag{1}$$

subjected to

$$\sum_{j=1}^{m} x_{ij} = 1, \quad i = 1, \ldots, n, \tag{2}$$

$$\sum_{i=1}^{n} w_{ij} x_{ij} \leq Y_j, \tag{3}$$

$$x_{ij} = 0 \text{ or } 1, \tag{4}$$

where $Y_j$ is the permissible maximum cost for $A_j$, $w_{ij}$ is the maximum cost when $T_i$ is assigned to $A_j$, and $\phi_j$ is the design objective function.[33] In the multiple robots system considered in this work, $\phi_j$ can be determined by the negotiation strategy in Section 4.2.

**Remark 1.** We assume that each robot is rational, i.e., it wants to maximize its expected utility. As a consequence, the objectives of the robots are conflicting and it is impossible to achieve local optimal for all the robots simultaneously. In this work, we are interested in finding the Pareto-optimal solutions satisfying (1) by treating multiple robots coordination as a multi-objective optimization problem. This notion of Pareto optimality is widely used in mathematical economics to model individual consumers striving to optimize distinct economic goals, and the Pareto solutions are the ones where there exist no solutions that are better for all robots in the negotiation.[34,35] It is noted that there is usually no single optimal solution for the multi-objective optimization, but a set of alternatives with different trade-offs. Despite the existence of multiple Pareto-optimal solutions, in practice, usually only one of these solutions is to be chosen. We use the Zeuthen negotiation strategy[36] to obtain the unique solution.

This paper will first present a contract net-based algorithm for the initial task allocation in Section 3 and then the task reallocation algorithm based on game theory in Section 4. The objective of task allocation is trying to achieve Pareto optimality.

## 3. Initial Task Allocation

In this section, we construct the initial task set for each robot following the marginal cost calculation-based contract net protocol in refs. [37, 38]. The nearest-neighbor insertion cost is used for the marginal cost and this marginal cost is used for the subsequent contract net protocol.

For each robot in negotiation, define the set of potential tasks as $\mathcal{T}_i = \{T_1, \ldots, T_{|\mathcal{T}|}\}$, and the cost performing its tasks as $C(\mathcal{T}_i)$. If a new task $T_g$ is added to $\mathcal{T}$, the marginal cost of the task $T_g$ can be defined as the incremental cost to complete this task, i.e.,

$$C_M^{\text{add}}(T_g|\mathcal{T}_i) = C(\mathcal{T}_i \cup \{T_g\}) - C(\mathcal{T}_i). \tag{5}$$

In contrast, when $T_g$ is removed from the task set, the marginal cost can be defined as the decremented cost for removing this task, i.e.,

$$C_M^{\text{remove}}(T_g|\mathcal{T}_i) = C(\mathcal{T}_i) - C(\mathcal{T}_i \backslash T_g), \tag{6}$$

where $C(\mathcal{T}_i \backslash T_g)$ is the cost for completing the set of tasks when $T_g$ is removed from $\mathcal{T}_i$.

There are two types of the individuals in the contract net-based mechanism, namely manager and contractor. The manager has a list of all the tasks information and announces the tasks to others in order to get bids from them. Each robot locally calculates its marginal cost for performing a set of tasks using (5). Then the task will be awarded to the robot owing the optimal bidding value, and this robot will put this task into its task set. The bid price for task $T_g$ of each robot in this work is calculated by

$$\text{Bid}_i(T_g) = C_M^{\text{add}}(T_g|\mathcal{T}_i) + C(\mathcal{T}_i), \quad i = 1 \ldots, m. \tag{7}$$

The algorithm for determining the initial task set of each robot using the contract net-based protocol is shown in Algorithm 1. It is noted that the contract net can be used for multiple robots negotiation and a local optimal solution can be guaranteed. However, the calculation of the marginal cost in (5) is handled in order of receiptor. Therefore, the roles of the robots in the negotiation are not equal. In order to optimize the efficiency of the whole system, we reallocate the tasks in the next section.

**Algorithm 1:** Initial Task Allocation

**Input**: $\mathcal{T} = \{T_1, \ldots, T_n\}$: A set contains all the tasks
**Output**: $\{\mathcal{T}_i, \ldots, \mathcal{T}_m\}$: Initial allocation of the tasks to the robots

```
forall the T_k ∈ T do
    Announcing T_k to all the robots, A_i, i = 1, ..., m
    if Bid_j(T_k) = max_{i=1,...,m} Bid_i(T_k) then
        T_j = T_j ∪ {T_k}
```

## 4. Tasks Reallocation

To improve the effectiveness of the multi-robot system, we use the game theory-based negotiation mechanism to reallocate the tasks to robots on the basis of the initial assignments in the previous section. There are two main parts in the negotiation, including negotiation protocol and the negotiation strategies.

### *4.1. Negotiation protocol*

#### *4.1.1. Determine negotiation tasks.*

**Definition 1.**[39] *Negotiation utility function (NUF)*: Define $\mathcal{T}_i$, $i = 1, \ldots, m$ as the task set for robot $A_i$, and $\delta$ as the negotiation solution, then the negotiation utility function for robot $A_i$ can be described as

$$U_i(\delta) = C(\mathcal{T}_i) - C_i(\delta). \tag{8}$$

The objective of the negotiation is to optimize the individual cost and the system cost. The negotiation deals can be defined by the NUF. There are two constraints for the negotiation solution:

- Assuming that the robots in negotiation are rational, i.e., the negotiation solution $\delta$ will not increase an extra cost for each individual, then the negotiation solution needs to satisfy $U_i(\delta) \geq 0$, $i = 1, \ldots, m$, i.e.,

$$(U_1(\delta), \ldots, U_m(\delta)) \succeq (0, \ldots, 0). \tag{9}$$

  This constraint is the rational constraint of the negotiation solution.
- As the objective of the negotiation is to satisfy the optimal solution of the whole system, the negotiation solution $\delta$ belongs to the so-called Pareto-optimal solution, i.e., $\forall \delta' \neq \delta$, the following inequations hold:

$$(U_1(\delta), \ldots, U_m(\delta)) \succeq (U_1(\delta'), \ldots, U_m(\delta')). \tag{10}$$

  This constraint is the constraint for the optimization.

Through the definition of these two constraints for the negotiation, the negotiation deals offer optimized individual and system costs. In order to reallocate the tasks to robots, we first propose an algorithm determining the tasks involved in the negotiation for each robot, i.e., constructing a negotiation task set $\mathcal{N}_S$ which consists of the tasks that will be included in the negotiation process. When robot $A_i$ negotiates with robot $A_j$, we compare the marginal costs of the tasks that belong to different robot individually and extract the negotiation task set from the initial assignment set $\mathcal{T}_i = \{T_{i(1)}, \ldots, T_{i(n)}\}$. The detailed procedure selecting tasks involved in the negotiation for robot $A_i$ is described in Algorithm 2.

**Algorithm 2:** Select Negotiation Tasks

**Input**: $\mathcal{T}_i$: Initial task allocation set of $A_i$
$A_j$: The robot negotiates with $A_i$
**Output**: $\mathcal{T}_{ij}$: Tasks for robot $A_i$ involved in negotiation with robot $A_j$

```
T_ij ← Ø
forall the T_g ∈ T_i do
    U_i(T_g) ← C_M^remove(T_g|T_i)
    Announces T_g to A_j and U_j(T_g) ← C_M^add(T_g|T_j)
    if U_i(T_g) > U_j(T_g) then
        T_ij ← T_ij ⋃ {T_g}
```

https://doi.org/10.1017/S0263574713000192 Published online by Cambridge University Press

*4.1.2. Determine negotiation robots.* In the multiple robots team, the robots organize themselves in a way that is mutually beneficial. Since the aggregate profit amassed by the individuals is directly tied to the success of the task, this self-organization yields the best results. Consider that a robot, $A_i$, shares its services with others. It does not force them to finish its work, but by convincing the group that they will make more profit by exchanging work than by acting individually or in subgroups. $A_i$ does this by investigating negotiation for utilizing all robots. If the group of robots comes up with a truly good negotiation together, it will maximize utility across the whole group. But there is a limit to this organization. As the group becomes larger, the combinatorics become intractable and the process of gathering all of the relevant information to produce a good negotiation becomes increasingly difficult. To overcome this problem, we propose an optimized way, which limits the simultaneous negotiation robots not more than three and optimizes the group utility via repeated negotiation.

To select negotiation robots, define an indicative function as

$$I(A_i, A_j) = \begin{cases} 1, & \text{if } \mathcal{T}_{ij} \neq \emptyset \text{ and } \mathcal{T}_{ji} \neq \emptyset, \\ 0, & \text{otherwise.} \end{cases} \qquad (11)$$

When the robot chooses the negotiation robots, it will randomly select one idle robot and check $I(A_i, A_j)$. If $I(A_i, A_j) = 1$, $A_i$ will negotiate with $A_j$. If the robot can choose two robots $A_j$ and $A_k$ satisfying $I(A_i, A_j) \cap I(A_i, A_k) \cap I(A_j, A_k) = 1$, then these three robots will negotiate among them. If any robot, $A_i$, cannot find any robot in the team satisfying $I(A_i, A_j) = 1,\ j \neq i$, it means that the system cannot be further optimized using the proposed negotiation mechanism. Then the negotiation process stops.

*4.1.3. Determine negotiation sets.* After the computation of the corresponding negotiation tasks set for robots $A_i$ and $A_j$, we select the negotiation solution set $\mathcal{N}_S$ satisfying the constraints from the task set $\bar{\mathcal{T}} = \mathcal{T}_{ij} \bigcup \mathcal{T}_{ji}$. Before computing the negotiation solution, we preprocess the tasks $T_g \in \bar{\mathcal{T}}$, calculating a utility of each task as

$$U(T_g) = \begin{cases} U_i(T_g) = C_M^{\text{remove}}(T_g|\mathcal{T}_i), & \text{if } T_g \in \mathcal{T}_i \\ U_j(T_g) = C_M^{\text{add}}(T_g|\mathcal{T}_i), & \text{if } T_g \in \mathcal{T}_j. \end{cases} \qquad (12)$$

Then, sort the tasks according to its utility in an ascending order as

$$U(T_1) \leq U(T_2) \leq \cdots \leq U(T_{|\bar{\mathcal{T}}|}). \qquad (13)$$

Define the task set of robots before negotiation by $\mathcal{T}_i$ and $\mathcal{T}_j$, and the initial task set in the negotiation by $\mathcal{T}_i^{\text{ini}} = \mathcal{T}_i \backslash \mathcal{T}_{ij}$, $\mathcal{T}_j^{\text{ini}} = \mathcal{T}_j \backslash \mathcal{T}_{ji}$, respectively. The initial negotiation task set is defined by $\mathcal{N}_S = \emptyset$, and the task sets in negotiation procedure are defined by $\mathcal{T}_i^{\text{T}}$ and $\mathcal{T}_j^{\text{T}}$, respectively. Let the initial values $\mathcal{T}_i^{\text{T}} = \mathcal{T}_j^{\text{T}} = \mathcal{T}_i^{\text{ini}}$. Based on the rational constraint and the optimal constraint, we use the branch-and-bound principle to solve the negotiation problem and provide the details in Algorithm 3.

**Algorithm 3:** NegotiationDealsConstruction

**Input**: $\bar{\mathcal{T}} = \mathcal{T}_{ij} \bigcup \mathcal{T}_{ji}$: A set of tasks involved in negotiation
$\mathcal{T}_i^{\text{T}}$: A set of tasks of robot $A_i$
$k$: An integer
**Output**: $\mathcal{N}_S$: A set of feasible negotiation solutions

1 $\mathcal{N}_S \leftarrow \emptyset$
2 **if** $k < |\bar{\mathcal{T}}|$ **then**
3   **forall the** $(\mathcal{T}_i^{\text{S}}, \mathcal{T}_j^{\text{S}}) \in \mathcal{N}_S$ **do**
4     **if** $U(\mathcal{T}_i^{\text{S}}, \mathcal{T}_j^{\text{S}}) \preceq U(\mathcal{T}_i^{\text{T}}, \mathcal{T}_j^{\text{ini}} \bigcup \bar{\mathcal{T}} \backslash (\mathcal{T}_i^{\text{T}} \backslash \mathcal{T}_i^{\text{ini}}))$ **then**
5       **return** false
6     **if** $U(\mathcal{T}_i^{\text{T}} \bigcup \{T_k\}) < U(\mathcal{T}_i)$ **then**
7       $\mathcal{T}_i^{\text{T}} \leftarrow \mathcal{T}_i^{\text{T}} \bigcup \{T_k\}$
8       NegotiationDealsConstruction($\bar{\mathcal{T}}, \mathcal{T}_i^{\text{T}}, k+1$)
9     **if** $U(\mathcal{T}_j^{\text{ini}} \bigcup \bar{\mathcal{T}} \backslash (\mathcal{T}_i^{\text{T}} \backslash \mathcal{T}_i^{\text{ini}})) < U(\mathcal{T}_j)$ **then**
10       NegotiationDealsConstruction($\bar{\mathcal{T}}, \mathcal{T}_i^{\text{T}}, k+1$)
11 **else**
12   $\mathcal{T}_j^{\text{T}} \leftarrow \mathcal{T}_j^{\text{ini}} \bigcup \bar{\mathcal{T}} \backslash (\mathcal{T}_i^{\text{T}} \backslash \mathcal{T}_i^{\text{ini}})$
13   $\mathcal{N}_S \leftarrow \mathcal{N}_S \bigcup (\mathcal{T}_i^{\text{T}}, \mathcal{T}_j^{\text{T}})$
14   **forall the** $(\mathcal{T}_i^{\text{S}}, \mathcal{T}_j^{\text{S}}) \in \mathcal{N}_S$ **do**
15     **if** $(U(\mathcal{T}_i^{\text{S}}), U(\mathcal{T}_j^{\text{S}})) \succeq (U(\mathcal{T}_i^{\text{T}}), U(\mathcal{T}_j^{\text{T}}))$ **then**
16       $\mathcal{N}_S \leftarrow \mathcal{N}_S \backslash (\mathcal{T}_i^{\text{S}}, \mathcal{T}_j^{\text{S}})$

To obtain the negotiation solution, we choose the upper bound from the utility function of the tasks of robots $A_i$ and $A_j$, and compare this value with the existing negotiation utility function. If this value is worse than the negotiation solution, i.e., violates the optimization constraint, then this branch will be deleted (lines 4 and 5); otherwise, construct a branch containing task $T_k$ in the state tree. Judge whether the rational constraint is satisfied when $T_k$ is put into $\mathcal{T}_i^{\text{T}}$ (lines 6–8); if true, construct a branch inserting $T_k$ in $\mathcal{T}_i^{\text{T}}$. Otherwise, if the potential new branch, $\mathcal{T}_i^{\text{T}}$, satisfies the constraints by inserting task $T_k$, construct this branch (lines 9 and 10).

Having judged all the negotiation tasks by a branch, the solution corresponding to the branch will be a potential negotiation solution (lines 12 and 13). We then check if these solutions satisfy the optimization constraints, by comparing each solution with the one in $\mathcal{N}_S$. If any $(\mathcal{T}_i^{\text{S}}, \mathcal{T}_j^{\text{S}}) \in \mathcal{N}_S$, $U(\mathcal{T}_i^{\text{S}}, \mathcal{T}_j^{\text{S}})$ is worse than $U(\mathcal{T}_i^{\text{T}}, \mathcal{T}_j^{\text{T}})$, then delete it from $\mathcal{N}_S$. While all the branches in the tree have been computed, the resulted set $\mathcal{N}_S$ will be the set of negotiation solutions.

It is noted that Algorithm 3 is a kind of the branch and bound algorithm, resulting in a large computational burden with large task numbers in $\mathcal{N}_S$. In order to decrease the computational complexity, it is better to reduce the numbers of tasks involved in the negotiation.

As each robot considered in this work is a rational individual, it tries to select the negotiation deal suitable to its optimization objective as the negotiation solution. It

https://doi.org/10.1017/S0263574713000192 Published online by Cambridge University Press

is however in the set of Pareto solutions that there are no solutions guaranteeing two robots approaching optimization simultaneously. The negotiation protocols should be determined, by which the robots will come to a consensus from the negotiation set $\mathcal{N}_S$. In this work, the receding concession is employed for the negotiation protocol,[40] which contains three main parts, including preprocessing the negotiation deals, determining the negotiation objectives and constructing the action set of negotiation.

*Preprocessing the negotiation deals.* Before negotiation, robot $A_i$ calculates the $U_i(\delta)$ for all the negotiation deals $\mathcal{N}_S$ and sorts these utilities in a descending order. The rational robot will choose the solution with larger utility, making its own utility optimal.

*Determining the negotiation objectives.* When robot $A_j$ receives a negotiation proposal $\delta_i \in \mathcal{N}_S$ from $A_i$, it will calculate $U_j(\delta_i)$. If this utility is not smaller than the utility of the negotiation proposal raised by itself, $U_j(\delta_i) \geq U_j(\delta_j)$, i.e., the negotiation proposal raised by others is better than its own proposal, we call this state as the negotiation objective. If the negotiation approaches the objective, robots will reallocate the tasks according to $\delta_i$.

*Constructing the action set of negotiation.* If the negotiation proposals made by $A_i$ and $A_j$ do not satisfy the negotiation objectives. Robots will make new proposals $\delta_i' \in \mathcal{N}_S$ satisfying $U_j(\delta_i') > U_j(\delta_i)$, which increases the utility of $A_j$. This proposal is called the receding proposal. As long as one robot makes the receding proposal, the robots will renegotiate. Otherwise, the negotiation will conclude a conflict, making it not being able to approach the mutual negotiation.

### *4.2. Negotiation strategies*

From the negotiation flow, we can find that it is important to define the negotiation strategy, i.e., the actions robots take if the negotiation proposal does not satisfy the negotiation objective. As the goal of negotiation is to optimize the cost performing tasks, the conflict deal cannot optimize this cost. In this work, we use the Zeuthen negotiation strategy[36] to avoid the negotiation conflict.

Robots $A_i$ and $A_j$ compute the Zeuthen risk value $\text{Risk}_i^\delta$ and $\text{Risk}_j^\delta$ according to its negotiation suggestion respectively using the following equation:

$$\text{Risk}_i^\delta = \begin{cases} 1, & U_{\delta_i} = 0 \\ \frac{U_i(\delta_i) - U_j(\delta_j)}{U_i(\delta_i)}, & U_{\delta_i} \neq 0. \end{cases} \quad (14)$$

If $\text{Risk}_i^\delta > \text{Risk}_j^\delta$, it will result in a conflict negotiation. Because of the smaller cost of $A_i$, $A_j$ will take the receding action and make a new negotiation proposal. If $\text{Risk}_i^\delta = \text{Risk}_j^\delta$, it will randomly choose a robot to take the receding action.

### *4.3. Scalability*

The proposed negotiation process works in a distributed way. For each agent, the computational complexity depends on two aspects: first, the selection of the negotiation robots and, second, the complexity of constructing the negotiation set, $\mathcal{N}_S$.

Let $\mathcal{A}$ contain $m$ robots and $\mathcal{T}$ contain $n$ tasks. Based on the proposed mechanism, the selection of negotiation robots may be determined in time $o(m^3)$ as there are at most three robots involved in one negotiation procedure.

The complexity of constructing the negotiation set, $\mathcal{N}_S$, depends on two factors: the number of robots involved in one negotiation procedure and the cardinal of the negotiation set, $|\mathcal{N}_S|$. If the number of robots involved in one negotiation procedure is not constrained, i.e., all the robots involve in, then the problem will be PSPACE-hard.[41] However, the proposed mechanism limits the number of robots in one negotiation procedure as two or three. Then, the maximum computational complexity will be $o(2^n)$ or $o(3^n)$. In this work, we employ the utility function to constrain the number of tasks involved in negotiations, making the computational cost decrease phenomenally. If the maximum number of the tasks calculated by using the utility function is fixed, the computational cost of the proposed negotiation protocol will be polynomial.

As there are two or three robots negotiating among themselves in one negotiation procedure, the bottleneck of the proposed methodology is that the negotiation in one procedure only achieves the local Pareto optimization for these two or three robots. Group utility needs multiple negotiation procedures to be optimized. For this reason, the proposed methodology is applicable to medium-size robot teams. In this case, the global optimization can be achieved through fewer negotiation procedures.

## 5. Simulation Study

In this section, we study the problem of allocating tasks to robots, where tasks are simply locations in a map that have to be visited by the robots. There are many possible connection costs for these targets, of which we have used the nearest insertion cost.[42] Hence, the marginal cost putting task $T_g$ to the task set $\mathcal{T}_i$ can be defined as

$$C_M^{\text{add}}(T_g|\mathcal{T}_i) = \min_{T_k \in \mathcal{T}_i}\{L(T_k, T_g) + L(T, T_{k+1}) - L(T_k, T_{k+1})\}, \quad (15)$$

and the marginal cost removing task $T_g$ from $\mathcal{T}_i$ can be defined as

$$C_M^{\text{remove}}(T_g|\mathcal{T}_i) = L(T_k, T_g) + L(T_g, T_{k+1}) - L(T_k, T_{k+1}), \quad (16)$$

where $L(\cdot)$ is the cost for completing the task. In the vehicle routing problem, one may use the direct cost $C_M^{\text{add}}(T_g|\mathcal{T}_i) = 2\min_{T_k \in \mathcal{T}_i}\{L(T_k, T)\}$ as the marginal cost adding target $T_g$. It provides an upper bound on the routing cost. The nearest insertion cost used in this work works well because it is usually accurate for small sets $\mathcal{T}_i$.

**Remark 2.** In the problem studied in the simulation, we can simply use the Euclidean distance as the moving cost. Other cost functions containing the motion constraints or the mission constraints of the robot can also be employed for specific applications such as emitting the threat targets using UAVs.

In the first simulation study, there are three robots in negotiation. Two scenarios containing 30 and 50 target

https://doi.org/10.1017/S0263574713000192 Published online by Cambridge University Press

Table I. Simulation results for 30 targets.

| Robot | Initial number of targets | Initial cost | Number of targets after negotiation | Cost after negotiation | Cost reduction ratio |
|---|---|---|---|---|---|
| 1 | 10 | 327.94 m | 8 | 240.01 m | 26.81% |
| 2 | 9 | 255.44 m | 9 | 207.64 m | 18.72% |
| 3 | 11 | 241.33 m | 13 | 234.62 m | 2.78% |

points are considered, respectively. As shown in Fig. 1, the targets are randomly distributed in a 100 m × 100 m area, which are denoted by '○'. The initial positions of the robots are $P_1 = [10\,\text{m}, 25\,\text{m}]$, $P_2 = [80\,\text{m}, 20\,\text{m}]$, and $P_3 = [50\,\text{m}, 80\,\text{m}]$, which are denoted by '■' in Fig. 1.

For each targets set, we first use the contract net-based mechanism to obtain the initial task allocation. The cost of each robot can be simply calculated as the moving distance after visiting all the targets. The initial allocation results using the contract net-based mechanism are shown in Figs. 1(a) and (b). Based on the initial target assignment, negotiation-based target reallocation results using our proposed mechanism are shown in Figs. 1(c) and (d), respectively. In the figures, we use lines with consistent color to connect the targets belonging to the same robot.

The costs for the task allocation problem are shown in Tables I and II, respectively. The cost reduction ratio of robot $A_i$ is defined as

$$\text{Ratio}_i = \frac{\text{Initial cost} - \text{Cost after negotiation}}{\text{Initial cost}} \times 100\%. \tag{17}$$

From Tables I and II, we can find that the costs of all robots are reduced after negotiation. Specifically, for the case where there are 30 targets, $A_1$ has the maximum cost reduction ratio, 26.81%, and $A_3$ has the minimum cost reduction ratio, 2.78%. For the case where there are 50 targets, $A_3$ has the maximum reduction cost ratio, 31.45%, and $A_1$ has the minimum reduction cost ratio, 11.93%. As each robot is

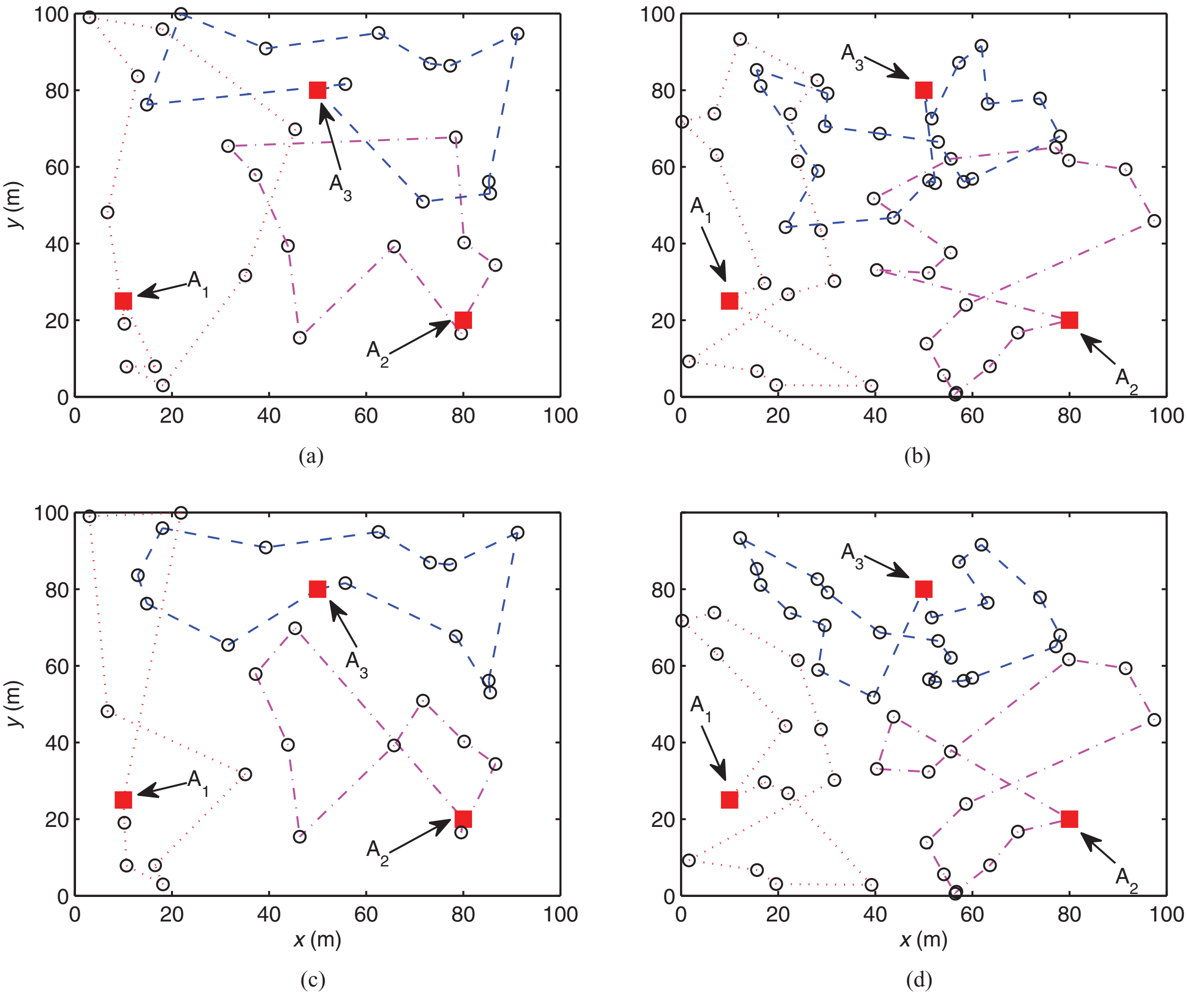



Fig. 1. (Colour online) Simulation results (three robots with 30 and 50 targets, respectively). (a) Initial target allocation for targets set 1 (30 targets). (b) Initial target allocation for targets set 1 (50 targets). (c) Target reallocation of targets set 1 (30 targets). (d) Target reallocation of targets set 2 (50 targets).

https://doi.org/10.1017/S0263574713000192 Published online by Cambridge University Press

Table II. Simulation results for 50 targets.

| Robot | Initial number of targets | Initial cost | Number of targets after negotiation | Cost after negotiation | Cost reduction ratio |
|---|---|---|---|---|---|
| 1 | 15 | 269.08 m | 13 | 236.97 m | 11.93% |
| 2 | 16 | 289.18 m | 14 | 241.62 m | 16.45% |
| 3 | 19 | 360.93 m | 23 | 247.41 m | 31.45% |

rational and wants to reduce its own cost, there are no robots that increase their cost.

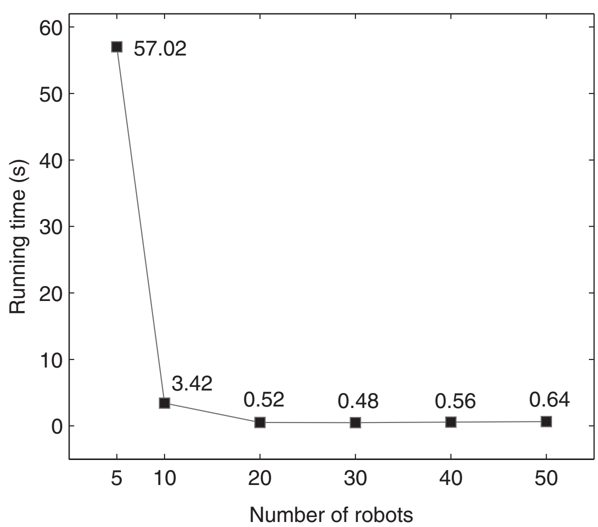
It is noted that from Fig. 1(c), $A_1$ still keeps tasks that are closer to the path of $A_3$. This is because in the task allocation process, we use the simple nearest insertion to compute the connection cost of the tasks, and we try to obtain the Pareto-optimal solution of the robots. A global optimal solution cannot be achieved using the decentralized mechanism proposed in this work.

In order to evaluate the proposed approach in more detail, we further ran two batches of simulations. In the first batch, the number of robots varies while the number of targets (tasks) is kept fixed. We set that there are 100 targets randomly distributed in a 1000 m × 1000 m square area, and consider different cases with 5, 10, 20, 30, 40 and 50 robots in the task allocation, respectively. In each case, it is assumed that the robots uniformly lie in the diagonal of the square.

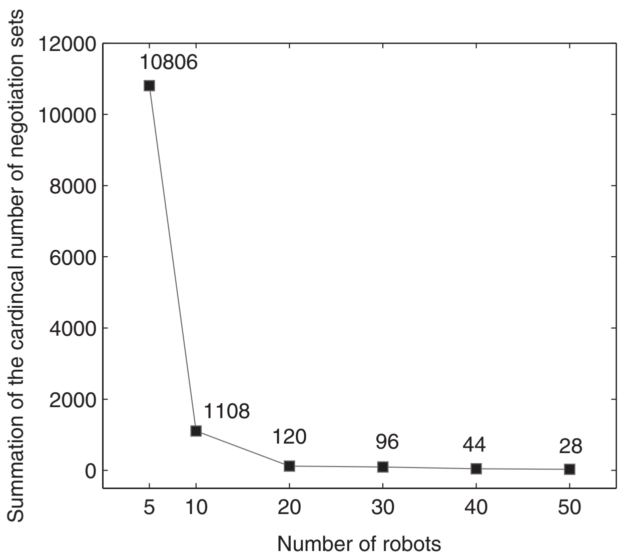
In the second batch of simulations, the number of targets (tasks) varies while the number of robots is kept fixed. We set that there are 10 robots which uniformly lie in the diagonal of a 1000 m × 1000 m square, and consider different cases with 50, 80, 100, 120, 150, 180 and 200 targets, respectively. It is assumed that all the targets are randomly distributed in the square for all the cases.

We mainly examine the cost reduction ratio and the running times when employing our algorithm for such cases. Simulation results for the first batch are shown in Figs. 2–4. From Fig. 2, we can find that the running time of the algorithm decreases as more robots involved in the task allocation. Specifically, when there are only five robots sharing 100 targets, the algorithm runs very slow compared with other cases with more robots. This is because if the number of robots is small while the number of targets is large, each robot has a large initial allocation target set and this initial allocation ensures that there are many targets that may be selected as the possible negotiation tasks in the reallocation procedure. As discussed in Sections 4.1.3 and 4.3, the construction of the negotiation set will be complex and the cardinal number of the negotiation set will be very large, as shown in Fig. 2(b). The numbers shown in Fig. 2(b) are the summation of the cardinal number of the negotiation set of all the robots. When the number of robots is large, the contract- based task allocation will provide a better initial task allocation solution, then the negotiation set will be small and the algorithm runs faster.

Figure 3 gives the statistics of the cost reduction ratio after the negotiation against the initial contract-based allocation. It is clear that average costs of the robots are all decreased after negotiation for all the cases. When the number of robots is large, we can find that the costs of some robots are not decreased. This is due to the fact that the number of targets initially allocated to the robots is small; therefore, it is possible that there are some robots that have no targets involved in negotiation or the negotiation result is the same as initial allocation. This can also be observed in Figs. 3(b) and 4. Figure 3(b) indicates that there are many robots that have




(a)




(b)

Fig. 2. Running times and the cardinal number of the negotiation sets for the cases with different number of robots. (a) Running times. (b) Summation of the cardinal number of the negotiation sets for all the robots.

https://doi.org/10.1017/S0263574713000192 Published online by Cambridge University Press

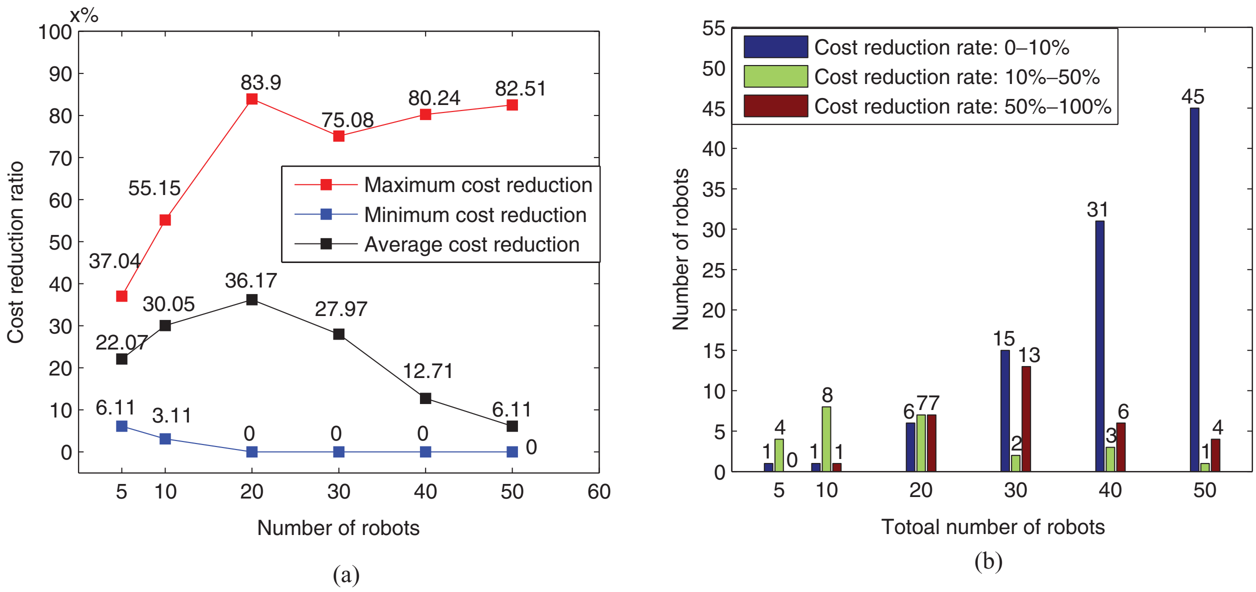



Fig. 3. (Colour online) Cost reduction of robots. (a) Cost reduction ratio of robots. (b) Statistics of number of robots that have cost reduction ratios within different ranges.

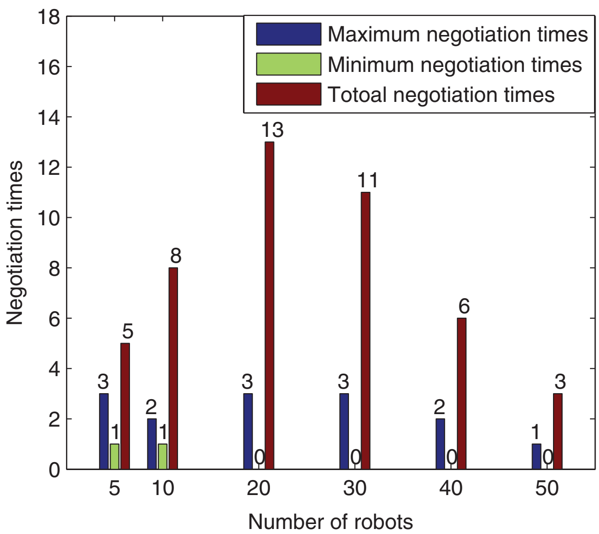



Fig. 4. (Colour online) Negotiation times for the cases with different number of robots.

the cost reduction ratio less than 10% in the case where a large number of robots involve in the task allocation. Figure 3(b) shows that the negotiation times will decrease when the number of robots increases. The maximum negotiation times decrease to 1 and the total negotiation times decrease to 3 when there are 50 robots sharing 100 targets. This is also due to the fact that the target set of the initial allocation is small for each robot and the chance of the targets that will be selected as the negotiation target is small.

Simulation results for the second batch of simulation are shown in Figs. 5–7. It is also noticed from Fig. 5 that the average traveling costs decrease after negotiation. When more targets are incorporated in the environment, the cost reduction will be more remarkable. The reason is that if more targets are added, each robot will have a large set of initial targets and the possible negotiation set increases. Then more negotiations should be done for the robots. This increasing of targets also increases the cardinal number of the negotiation set and makes the program run slower as shown in Fig. 6. Figure 7 also indicates that the negotiation times increase as the number of targets increases. It is noted that the results are also affected by the distribution of robots and targets in the environment.

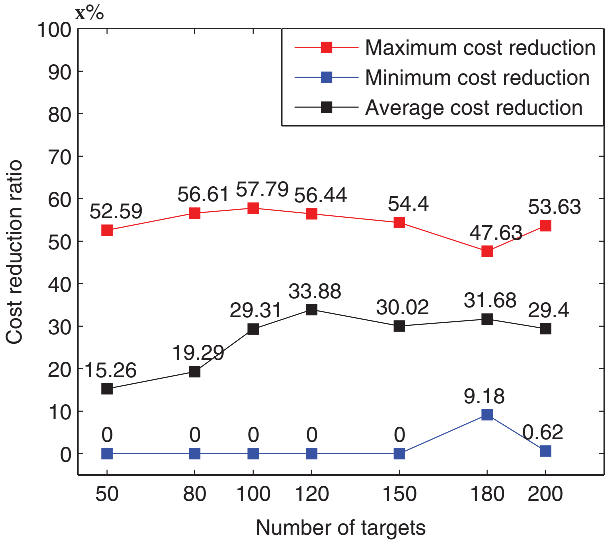
In summary, these preliminary simulations indicate that the negotiation mechanism provides a computationally efficient approach to the problem of task allocation. Compared with the contract-based initial task allocation, the traveling cost of the robots is decreased. As shown in the simulation results discussed in Section 4.3, our algorithm is affected by the number of targets and robots. The running time will increase when the number of targets increases. As our negotiation mechanism works in a decentralized manner and we constrain the number of negotiation objects (robots), the algorithm can also work well when the number of robots increases. These results have shown that the game theory-based negotiation offers advantages over the simple contract-based task allocation. In the scenarios given, performance has been improved. If there are few tasks entering the system, some tasks can be effectively allocated using the initial contract-based task allocation. If there are many tasks, the negotiation-based reallocation is needed to reduce the cost.

## 6. Experimental Results

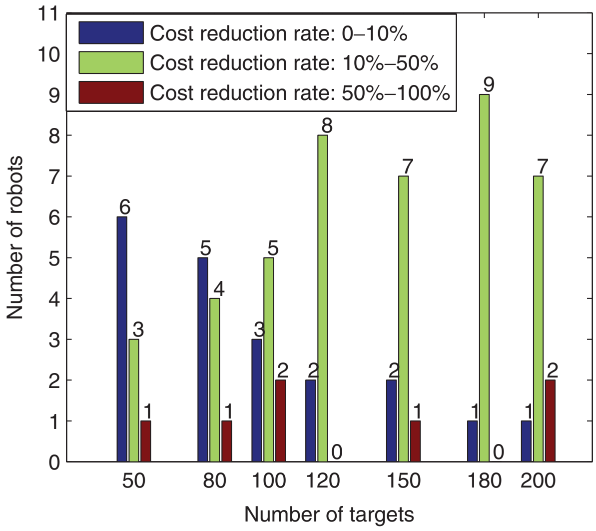
In the experiment, we use the proposed negotiation mechanism for the target assignment of three Pioneer $3^{TM}$-DX (referred to as P3dx) robots as shown in Fig. 8. The negotiation algorithm is developed in the VC.net environment.

Because of the lack of independent communication ability on the P3dx robot, the communications among the robots

https://doi.org/10.1017/S0263574713000192 Published online by Cambridge University Press

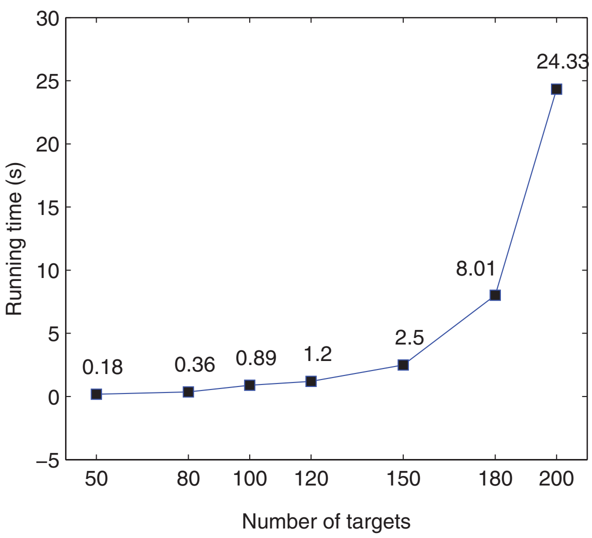



(a)

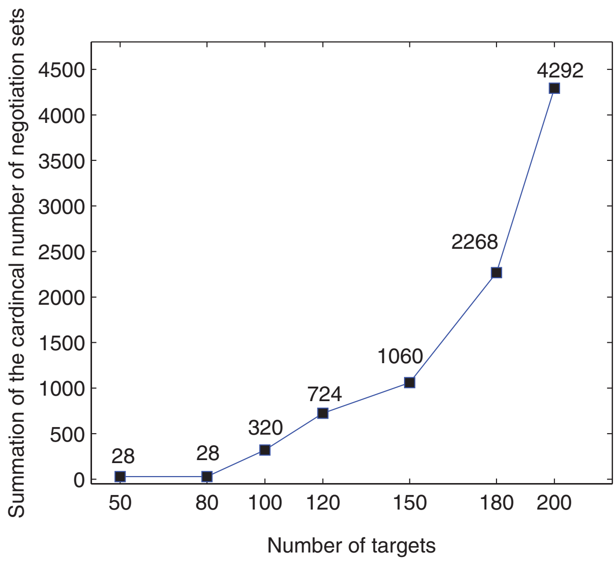



(b)

Fig. 5. (Colour online) Cost reduction of robots. (a) Cost reduction ratio. (b) Statistics of number of robots that have cost reduction ratios within different ranges.




(a)




(b)

Fig. 6. (Colour online) Running times and the cardinal number of the negotiation sets for the cases with different number of targets. (a) Running times. (b) Summation of the cardinal number of the negotiation sets for all the robots.

are realized via the data packet `ArPacket` provided in the Advanced Robotics Interface for Applications (ARIA)[1]. To prevent the drop of communication, we build the Server and Client program for every robot, where the Server program is used to receive the data which are sent from a robot (including the robot itself and others), and the Client is used to process and send the data. To verify the multiple robot negotiation mechanism, we divide the experiment into two parts, namely allocation and negotiation individually.

[1] ARIA is a software provided by MobileRobots, Inc., USA, for the Pioneer robots. Users can develop their own algorithms based on the existing classes and interfaces provided by the software.

### *6.1. Implementation of task allocation*

The system structure of the task allocation system in the experiment is designed as shown in Fig. 9. In the experiment, there are totally 11 targets in the environment, which are described as the ID$(x,y)$ style as $1(-2, 2), 2(-7, 7), 3(-5, 5), 4(-6, 6), 5(1.2, 12), 6(3, 3), 7(2, 2), 8(-7, -7), 9(-3, -3), 10(-4, -4), 11(-6, -6)$.

The task allocation system for the robot consists of two parts; one is to allocate the task and another is to obtain the task. In the negotiation process, each robot allocates its targets set to other robots and selects the robot to which it will allocate the targets according to the position of the robot. Table III shows the target assignment process of each robot. It is shown that the contract net-based approach can guarantee that each target is assigned to the nearest robot.

https://doi.org/10.1017/S0263574713000192 Published online by Cambridge University Press

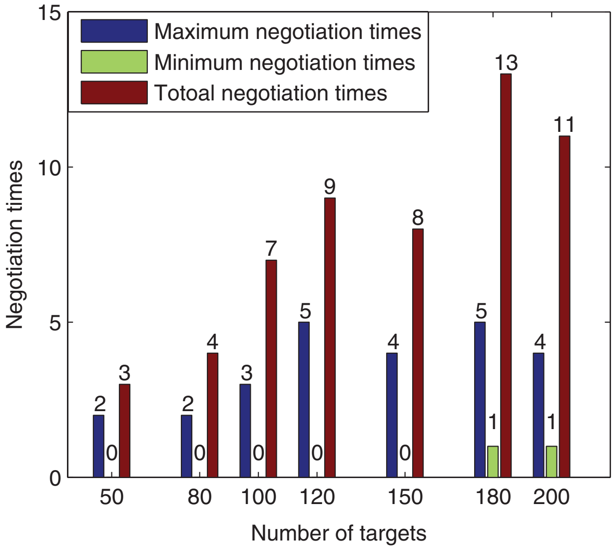



Fig. 7. (Colour online) Negotiation times for the cases with different number of targets.

Fig. 8. (Colour online) Pioneer 3™-DX robots.

*A. Selecting the negotiation robot*
If there are several robots in idle process, the robot will take high priority to select the robot which has the minimum negotiation times with the current robot. When the negotiation times of two robots are the same, the robot will select the robot with lower ID as its negotiation object. The framework of selecting the negotiation object is shown in Fig. 10. There are three robots involved in the experiment,

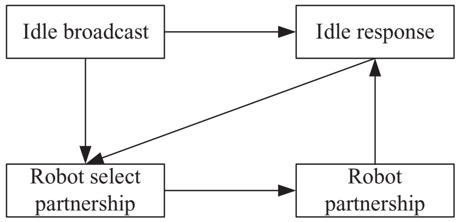



Fig. 10. Framework of selecting the negotiation object.

and the maximum negotiation times of each robot are set as 5.

In the experiment, each robot selects the negotiation robot in accordance with the negotiation times and the robot ID. If there are two robots that select each other as the negotiation robot, then we can confirm one selection of the negotiation object and display the negotiation object and negotiation times in Table IV. We repeat the negotiation object selection process until the negotiation times between all the robots satisfy the predefined requirements. We can find that the negotiation system satisfies the predefined negotiation object selection policy.

*B. Negotiation between robots*
To ease the experiment, the negotiation is a pair-wised negotiation based on the pre-specified negotiation object. Each robot has its own targets set and the negotiation will obtain the optimal allocation of these targets. Before the negotiation, each robot announces its targets set to its negotiation partner. Then, each robot computes the negotiation proposal solution based on the targets set of its partner and itself and then announces it to its partner. Each robot will have the proposal solutions of the robot itself and the one received from its partner, individually. According to the utility function, the robot will choose the solution which makes the whole utility function optimal. In this experiment, $A_2$ proposes a solution which is accepted by both robots. Robots will exchange the target based on the negotiation solution. The targets set of each robot, including the ID, position and the angle information, before and after negotiation is shown in Table 7. We can find that the solution after negotiation is better than the initial task allocation solution.

## 7. Conclusion
In this work, we have investigated the game theory-based negotiation methodology for multiple robots task allocation. The basic contract net-based approach has been employed to obtain the initial task allocation. A new method to select the negotiation robots and constructing the negotiation set has

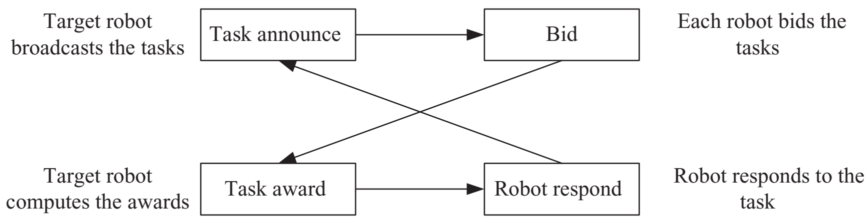



Fig. 9. Framework of the task assignment.

https://doi.org/10.1017/S0263574713000192 Published online by Cambridge University Press

Table III. Initial task allocation process.

| | | |
|---|---|---|
| Client connected to server | Client connected to server | Client connected to server |
| $A_1$ win the auction of #1 | $A_2$ win the auction of #5 | $A_3$ win the auction of #9 |
| $A_1$ win the auction of #2 | $A_2$ win the auction of #6 | $A_3$ win the auction of #7 |
| $A_1$ win the auction of #3 | $A_2$ finishes allocation. | $A_3$ win the auction of #10 |
| $A_1$ win the auction of #4 | | $A_3$ win the auction of #11 |
| $A_1$ win the auction of #8 | | $A_3$ finishes allocation. |
| $A_1$ finishes allocation. | | |

Table IV. Process of selecting negotiation robot.

| Robot $A_1$: | Robot $A_2$: | Robot $A_3$: |
|---|---|---|
| Connected to server | Connected to server | Connected to server |
| deal with $A_2$ win, 0 times | deal with $A_3$ win, 0 times | deal with $A_2$ win, 0 times |
| deal with $A_2$ win, 1 times | deal with $A_3$ win, 1 times | deal with $A_2$ win, 1 times |
| deal with $A_3$ win, 0 times | deal with $A_1$ win, 0 times | deal with $A_1$ win, 0 times |
| deal with $A_3$ win, 1 times | deal with $A_1$ win, 1 times | deal with $A_1$ win, 1 times |
| deal with $A_2$ win, 2 times | deal with $A_1$ win, 2 times | deal with $A_2$ win, 2 times |
| deal with $A_3$ win, 2 times | deal with $A_3$ win, 2 times | deal with $A_2$ win, 3 times |
| deal with $A_2$ win, 3 times | deal with $A_3$ win, 3 times | deal with $A_2$ win, 4 times |
| deal with $A_3$ win, 3 times | deal with $A_3$ win, 4 times | deal with $A_1$ win, 2 times |
| deal with $A_2$ win, 4 times | deal with $A_1$ win, 3 times | deal with $A_1$ win, 3 times |
| deal with $A_3$ win, 4 times | deal with $A_1$ win, 4 times | deal with $A_1$ win, 4 times |
| deal with $A_2$ win, 5 times | deal with $A_1$ win, 5 times | deal with $A_1$ win, 5 times |
| deal with $A_3$ win, 5 times | deal with $A_3$ win, 5 times | deal with $A_2$ win, 5 times |

Table V. Targets of each robot.

| Robot | Position | Targets ID (initial allocation) | Targets ID after negotiation |
|---|---|---|---|
| 1 | (1, 1) | 1, 2, 3, 4, 8 | 7, 5, 6 |
| 2 | (−1, 1) | 5, 6 | 1, 2, 3, 4 |
| 3 | (−1, −1) | 9, 7, 10, 11 | 9, 10, 11, 8 |

been proposed by employing the utility functions. Next, a negotiation mechanism that is suitable for the decentralized task allocation has been presented. Subsequently, a game theory-based negotiation strategy is proposed to achieve the unique Pareto-optimal solution for the multi-robots task allocation problem. Extensive simulations have illustrated that the task allocation solutions after the negotiation are better than the initial contract-based task allocation. The impact of the number of robots and tasks has also been provided through the simulation study. It has been shown that our algorithm can work well with medium sizes of robots and tasks.

Future research directions include: (i) we need to optimize the negotiation set construction procedure for specific problems, and decrease the computational complexity in this procedure; (ii) we may employ the framework of graphical games, such as the one used in our previous work,[43] to make the algorithms more flexible, and be suitable for fault tolerant task allocation when some robots are broken; (iii) we will extend our algorithms so that constrained and tight tasks can be handled. A typical example for constrained are two tasks that cannot be done independently by a single robot. Tight tasks cannot be decomposed into further single tasks. In this case, a subgroup of robots could determine their joint marginal costs and submit joint bids for such type of tasks. Also, to ensure cost independence between the subteams, the proposed framework should be extended to include the constraint that the subteams being awarded tight tasks are disjoint. In addition, the negotiation will be done between the subgroups rather than robot pairs.

## Acknowledgments

The authors would like to thank the editor and anonymous reviewers for their constructive comments that helped improve the quality and presentation of this paper. This work was supported by the National Natural Science Foundation of China under grant 51209174, the State Key Laboratory of Robotics and System (HIT) under grant SKLRS-2012-MS-04, and the Fundamental Research Program of Northwestern Polytechnical University (NPU) under grant JCY20130113.

The first two authors contributed equally to this work as the first author.

## References

1. C. Yang, Z. Li and J. Li, "Trajectory planning and optimized adaptive control for a class of wheeled inverted pendulum vehicle models," *IEEE Trans. Cybern.* **43**(1), 24–36 (2013).
2. C. H. Fua and S. S. Ge, "COBOS: Cooperative backoff adaptive scheme for multirobot task allocation," *IEEE Trans. Robot.* **21**(6), 1168–1178 (2005).
3. T. S. Dahl, M. Matarić and G. S. Sukhatme, "Multi-robot task allocation through vacancy chain scheduling," *Robot. Auton. Syst.* **57**, 674–687 (2009).
4. Z. Li, P. Y. Tao, S. S. Ge, M. Adams and W. S. Wijesoma, "Robust adaptive control of cooperating mobile manipulators with relative motion," *IEEE Trans. Syst. Man Cybern.* B **39**(1), 103–116 (2009).
5. C. Yang, G. Ganesh, S. Haddadin, S. Parusel, A. Albu-Schaeffer and E. Burdet, "Human-like adaptation of force and impedance in stable and unstable interactions," *IEEE Trans. Robot.* **27**(5), 918–930 (2011).

https://doi.org/10.1017/S0263574713000192 Published online by Cambridge University Press

6. R. Cui, S. S. Ge, V. E. B. How and Y. S. Choo, "Leader–follower formation control of underactuated autonomous underwater vehicles," *Ocean Eng*. **37**(17–18), 1491–1502 (2010).
7. Z. Li, X. Cao and N. Ding, "Adaptive fuzzy control for synchronization of nonlinear teleoperators with stochastic time-varying communication delays," *IEEE Trans. Fuzzy Syst*. **19**(4), 745 (2011).
8. Z. Li, J. Li and Y. Kang, "Adaptive robust coordinated control of multiple mobile manipulators interacting with rigid environments," *Automatica* **40**(12), 2028–2034 (2010).
9. M. J. Matarić, G. S. Sukhatme and E. H. Østergaard, "Multi-robot task allocation in uncertain environments," *Auton. Robots* **14**(2), 255–263 (2003).
10. M. Elango, S. Nachiappan and M. K. Tiwari, "Balancing task allocation in multi-robot systems using *k*-means clustering and auction based mechanisms," *Expert Syst. Appl*. **38**, 6486–6491 (2011).
11. B. Gerkey and M. J. Matarić, "A formal framework for the study of task allocation in multi-robot systems," *Int. J. Robot. Res*. **23**(9), 939–954 (2004).
12. H. Sayyaadi and M. Moarref, "A distributed algorithm for proportional task allocation in networks of mobile agents," *IEEE Trans. Autom. Control* **56**, 405–410 (2011).
13. T. C. Service and J. A. Adams, "Coalition formation for task allocation: Theory and algorithms," *Auton. Agent and Multi-Agent Syst*. **22**, 225–248 (2011).
14. Z. S. H. Chan, L. Collins and N. Kasabov, "An efficient greedy *k*-means algorithm for global gene trajectory clustering," *Expert Syst. Appl*. **30**, 137–141 (2006).
15. C. H. Cheng, G. W. Cheng and J. W. Wang, "Multi-attribute fuzzy time series method based on fuzzy clustering," *Expert Syst. Appl*. **34**, 1235–1242 (2008).
16. M. Strens and N. Windelinckx, "Combining planning with reinforcement learning for multi-robot task allocation," **In:** *Adaptive Agents and Multi-Agent Systems II* (D. Kudenko, D. Kazakov and E. Alonso, eds.), Springer, Berlin (2005) pp. 260–274.
17. S. Abdallah and V. Lesser, "Learning the Task Allocation Game," *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems*, ACM (2006) pp. 850–857.
18. K. G. Vamvoudakis and F. L. Lewis, "Multi-player non-zero-sum games: Online adaptive learning solution of coupled Hamilton–Jacobi equations," *Automatica* **47**(8), 1556–1569 (2011).
19. M. B. Dias, R. Zlot, N. Kalra and A. Stentz, "Market-based multirobot coordination: A survey and analysis," *Proc. IEEE* **94**(7), 1257–1270 (2006).
20. A. M. Khamis, A. M. Elmogy and F. O. Karray, "Complex task allocation in mobile surveillance systems," *J. Intell. Robot. Syst*. **64**, 33–55 (2011).
21. H.-L. Choi, L. Brunet and J. P. How, "Consensus-based decentralized auctions for robust task allocation," *IEEE Trans. Robot*. **25**, 912–926 (2009).
22. E. Rasmusen, *Games and Information: An Introduction to Game Theory*, Blackwell, Malden, MA (2007).
23. M. R. Adler, A. B. Davis, R. Wehmayer and R. W. Worrest, "Conflict Resolution Strategies for Nonhierarchical Distributed Agents," **In**: *Distributed Artificial Intelligence* (M. Huhns, ed.), Morgan Kaufmann, San Francisco, CA (1989) vol. 2, pp. 139–161.
24. S. Botelho, R. Alami and T. LAAS-CNRS, "M+: A Scheme for Multi-robot Cooperation Through Negotiated Taskallocation and Achievement," *IEEE International Conference on Robotics and Automation*, Detroit, MI (1999) vol. 2, pp. 1234–1239.
25. R. Davis and R. G. Smith, "Negotiation as a metaphor for distributed problem solving," *Artif. Intell*. **20**(1), 63–109 (1983).
26. E. H. Durfee, "Distributed Problem Solving and Planning," **In**: *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence* (W. Gerhard, ed.), MIT Press, Cambridge, Massachusetts, USA (1999) pp. 121–164.
27. T. Mullen and M. Wellman, "Some issues in the design of market-oriented agents," **In:** *Intelligent Agents II Agent Theories, Architectures, and Languages* (W. Michael, M. Jörg and T. Milind, eds.), Springer, Berlin (1996) pp. 283–298.
28. M. P. Wellman, "Market-oriented Programming: Some Early Lessons," **In**: *Market-based Control: A Paradigm for Distributed Resource Allocation* (S. Clearwater, ed.), World Scientific, Singapore (1996) pp. 74–95.
29. G. Zlotkin and J. S. Rosenschein, "Cooperation and conflict resolution via negotiation among autonomous agents in noncooperative domains," *IEEE Trans. Syst. Man Cybern*. **21**(6), 1317–1324 (1991).
30. T. Basar and G. J. Olsder, *Dynamic Noncooperative Game Theory*, SIAM Series in Classics in Applied Mathematics, SIAM, Philadelphia, PA (1999).
31. K. Sycara, S. F. Roth, N. Sadeh and M. S. Fox, "Distributed constrained heuristic search," *IEEE Trans. Syst. Man Cybern*. **21**(6), 1446–1461 (1991).
32. G. Zlotkin and J. S. Rosenschein, "Cooperation and conflict resolution via negotiation among autonomous agents in noncooperative domains," *IEEE Trans. Syst. Man Cybern*. **21**(6), 1317–1324 (1994).
33. G. P. Liu, J. B. Yang and J. F. Whidborne, *Multiobjective Optimisation and Control* (Research Studies Press, Baldock, Hertfordshire, UK, 2004).
34. I. Das and J. Dennis, "Boundary intersection: A new method for generating pareto optimal points in multicriteria optimization problems," *SIAM J. Optim*. **8**(3), 631–657 (1998).
35. R. Cui, B. Gao and J. Guo, "Pareto-optimal coordination of multiple robots with safety guarantees," *Auton. Robots* **32**(3), 189–205 (2012).
36. J. C. Harsanyi, *Rational Behavior and Bargaining Equilibrium in Games and Social Situations* (Cambridge, Cambridge University Press, 1977).
37. T. W. Sandholm, "An Implementation of the Contract Net Protocol Based on Marginal Cost Calculations," *Proceedings of the National Conference on Artificial Intelligence*, Washington, DC (1993) pp. 256–256.
38. T. W. Sandholm, "Contract Types for Satisficing Task Allocation: I. Theoretical Results," **In:** *AAAI Spring Symposium Series: Satisficing Models*, AAAI, California, USA (1998), pp. 68–75.
39. R. K. Ahuja, T. L. Magnanti, J. B. Orlin and K. Weihe, *Network Flows: Theory, Algorithms, and Applications* (Prentice Hall, Englewood Cliffs, NJ, 1993).
40. J. S. Rosenschein and G. Zlotkin, *Rules of Encounter* (MIT Press, Cambridge, MA, 1994).
41. T. Cormen, C. Leiserson, R. Rivest and C. Stein, *Introduction to Algorithms* (MIT Press, Cambridge, MA, 2001).
42. J. Bramel and D. Simchi-Levi, "A location based heuristic for general routing problems," *Oper. Res*. **43**(4), 649–660 (1995).
43. J. Guo, B. Gao, R. Cui and S. S. Ge, "Estimating the Minimum Number of Robots to Finish Given Multi-objects Task," *IEEE 5th International Conference on Cybernetics and Intelligent Systems (CIS)*, Qingdao, P. R. China (2011) vol. 1, pp. 170–174.