Proceedings of the 2017 International Workshop on Research, Education and Development on Unmanned Aerial Systems (RED-UAS 2017), 3-5 October, Linkoping, Sweden.
DOI:10.1109/RED-UAS.2017.8101660 ISBN:9781538609392

# A Game-theoretical Approach to Heterogeneous Multi-Robot Task Assignment Problem with Minimum Workload Requirements

Inmo Jang, Hyo-Sang Shin, and Antonios Tsourdos

***Abstract*— This paper addresses a multi-robot task assignment problem with heterogeneous agents and tasks. Each task has a different type of minimum workload requirement to be accomplished by multiple agents, and the agents have different work capacities and costs depending on the tasks. The objective is to find an assignment that minimises the total cost of assigned agents while satisfying the requirements of the tasks. We formulate this problem as the minimisation version of the generalised assignment problem with minimum requirements (MinGAP-MR). We propose a distributed game-theoretical approach in which each selfish player (i.e., robot) wants to join a task-specific coalition that minimises its own cost as possible. We adopt tabu-learning heuristics where a player penalises its previously chosen coalition, and thereby a Nash-stable partition is always guaranteed to be determined. Experimental results present the properties of our proposed approach in terms of suboptimality and algorithmic complexity.**

## I. Introduction

Cooperation of a huge number of small-sized autonomous aerial robots, called *UAV swarms*, will play a major role in complex missions that existing operational concepts using a few large UAVs could not deal with [1]. Even if each single agent in a swarm is incapable of accomplishing a task alone, their cooperation will lead to successful outcomes because of their robustness and adaptiveness. The possible applications include environmental monitoring [2], ad-hoc network relay [3], cooperative radar jamming [4], to name a few.

One of the main technical challenges for utilisation of a swarm is collective decision-making such as *multi-robot task assignment (MRTA) problem* [5]. This paper addresses a particular case in which each task can not be fulfilled by a single agent but requires multiple agents to be completed: the task's minimum workload requirement should be met by the total work capacities provided by multiple agents. This case falls into ST-MR category [5]. Moreover, we take into account heterogeneous tasks and agents: each task needs a different type of workload (e.g., task $t_1$ requires sensing ability, whereas task $t_2$ demands transportation capability); and each agent also, depending on the tasks, has different work capacities (or efficiencies) and costs. The objective is to find an assignment, in a distributed manner, that satisfies the requirements of all the tasks while minimising the aggregated cost of assigned agents. We formulate this problem as *the minimisation version of the generalised assignment problem with minimum requirements (MinGAP-MR)*, which is defined in Section II.

The (standard) generalised assignment problem (GAP) has been extensively studied over several decades. A general overview of GAP is available in [6] along with its real-life applications such as scheduling, transporting, and facility location. One of its key differences from MinGAP-MR is that GAP has maximum-capacity constraints for knapsacks instead of minimum requirements. In multi-robot system domain, a single agent has been typically considered as capable of executing multiple tasks (i.e., MT-SR case [7], [8]) within its work capacity limitation. Hence, this type of MRTA problems can be modelled as GAP or its variants [9]–[11], where each robot is regarded as a knapsack with its maximum-available size and tasks are items to insert.

On the contrary, few works in the literature consider minimum-requirement constraints for knapsacks. Even for a single knapsack problem, approximation algorithms based on greedy heuristics are relatively recently proposed in [12], [13]. For multiple-knapsack cases, the works in [14], [15] study assignment problems of students to lectures where there is the minimum number of participants required for each lecture to launch. However, this constraint is only concerned with the cardinality of assigned agents, and thereby it is not suitable for a problem case with heterogeneous agents. Although [16] considers such agents along with knapsack's minimum requirements as well as maximum limitations, the suggested approximation algorithm does not always provide a feasible solution (i.e., the resultant assignment often violates the knapsack constraints).

This paper proposes a game-theoretical approach for MinGAP-MR. As inspired by *coalitional games* [17, Chap 7], we regard each robot as a selfish player (i.e., item) who wants to join a task-specific coalition (i.e., knapsack) that minimises its cost. The objective of this game is to find a *Nash-stable partition*, i.e., a set of disjoint coalitions in which no agent will unilaterally deviate. To avoid possible conflicts between the players, we adopt a self-learning scheme by which every player gradually penalises its previously chosen coalition. In this paper, we prove that the proposed approach always determines a Nash-stable partition, and then investigate its algorithmic complexity and suboptimality through various experimental results. To the best of our knowledge, this paper is the first work that proposes a distributed approach for MinGAP-MR and utilises it for the MRTA problem considered.

Inmo Jang, Hyo-Sang Shin, and Antonios Tsourdos are with Centre for Autonomous and Cyber-Physical Systems, Cranfield University, MK430AL, UK. Email: {inmo.jang, h.shin, a.tsourdos}@cranfield.ac.uk. The authors gratefully acknowledge that this research was supported by International Joint Research Programme with Chungnam National University (No. EFA3004Z).

## II. Problem Formulation

This section formally defines MinGAP-MR. Suppose that there exist a set of $n_a$ agents $\mathcal{A} = \{a_1, ..., a_{n_a}\}$ and a set of $n_t$ tasks $\mathcal{T} = \{t_1, ..., t_{n_t}\}$. For each task $t_j$, each agent $a_i$ is associated with different work capacity (or efficiency) $w_{ij}$ and cost $c_{ij}$. Each task $t_j$ has its minimum workload requirement $R_j$ to be fulfilled by the aggregated work capacities of multiple agents. Any of the agents is incapable of executing any task alone (i.e., $\max_{\forall a_i \in \mathcal{A}} w_{ij} < R_j$, $\forall t_j \in \mathcal{T}$). Note that $R_j$, $w_{ij}$ and $c_{ij}$ are non-negative. The number of the agents $n_a$ is assumed to be sufficiently large so that there might be excessive agents who are not needed for the requirements. The objective is to distribute the agents to the tasks in a way that satisfies the demands of the tasks while minimising the total cost of assigned agents:

$$\min_{\{x_{ij}\}} \quad \sum_{i=1}^{n_a} \sum_{j=1}^{n_t} c_{ij} x_{ij} \tag{1}$$

$$\text{s.t.} \quad \sum_{j=1}^{n_t} x_{ij} \leq 1, \quad \forall i = 1, 2, ..., n_a \tag{C1}$$

$$\sum_{i=1}^{n_a} w_{ij} x_{ij} \geq R_j, \quad \forall j = 1, 2, ..., n_t \tag{C2}$$

$$x_{ij} \in \{0, 1\} \quad \forall i, j \tag{C3}$$

where $x_{ij} = 1$ if agent $a_i$ is assigned to task $t_j$. Constraint (C1) indicates that each agent can be exclusively allocated to at most one task. (C2) ensures that, for each task, the total work capacities provided by its assigned agents satisfy the minimum requirement. In this paper, an instance of MinGAP-MR is denoted by a tuple $(\mathcal{A}, \mathcal{T})$. MinGAP-MR is $\mathcal{NP}$-hard because it is a generalised version of the 0/1 minimisation knapsack problem (Definition 2), which is also the case [12].

TABLE I
Nomenclature

| Symbol | Description |
|---|---|
| $\mathcal{A}$ | A set of $n_a$ agents, i.e., $\{a_1, a_2, ..., a_{n_a}\}$ |
| $\mathcal{T}$ | A set of $n_t$ tasks, i.e., $\{t_1, t_2, ..., t_{n_t}\}$ |
| $w_{ij}$ | The work capacity of agent $a_i$ with regard to task $t_j$ |
| $c_{ij}$ | The (original) cost of agent $a_i$ to perform task $t_j$ |
| $\tilde{c}_{ij}$ | The learnt cost of agent $a_i$ to perform task $t_j$ (Eqn. (3)) |
| $\lambda$ | The learning rate to affect $\tilde{c}_{ij}$ (Eqn. (3)) |
| $R_j$ | The minimum requirement for task $t_j$ |
| $\Pi[\tau]$ | The (disjoint) partition of $\mathcal{A}$ at iteration $\tau$ |
| $\Pi[\tau](i)$ | The index of the task assigned to agent $a_i$, given $\Pi[\tau]$ |
| $S_j$ | The (task-specific) coalition for task $t_j$ |
| $e_i(S_j)$ | The expense of agent $a_i$ for coalition $S_j$ (Eqn. (2)) |

## III. A Game-theoretical Approach

### A. Preliminaries

Given an instance $(\mathcal{A}, \mathcal{T})$ of MinGAP-MR, we define a *partition* as a set $\Pi = \{S_1, ..., S_{n_t}, S_0\}$ such that $S_j \subseteq \mathcal{A}$, $\forall t_j \in \mathcal{T} \cup \{t_0\}$ and $S_j \cap S_k = \emptyset$ for $j \neq k$. Here, $t_0$ is *void task* indicating that "not to work any task in $\mathcal{T}$". $S_j$ is the (task-specific) *coalition* of agents assigned to task $t_j$. As the partition is iteratively evolved in our proposed approach, we use $\Pi[\tau]$ to denote the partition at iteration $\tau$. Given the partition, $\Pi[\tau](i)$ indicates the index of the task to which agent $a_i$ is assigned. Let $e_i(S_j)$, which will be defined in Section III-B, denote the *expense* for agent $a_i$ to execute task $t_j$ with other coworkers in $S_j$.

We model the problem considered as a coalitional game where every agent selfishly seeks to form a coalition that minimises its expense. Note that the expense is not the costs $c_{ij}$ but the virtual currency of the game to coordinate the self-interested agents. The key idea of our proposed approach is that, given the partition $\Pi[\tau]$, an arbitrary agent $a_i$ investigates the expenses for all possible coalitions (i.e., $e_i(S_j)$, $\forall S_j \in \Pi[\tau]$) and joins the most preferred one. The agent updates the partition information to $\Pi[\tau + 1]$, which reflects the agent's new decision, and broadcasts it to other agents. Another agent executes the same procedure at the next iteration, and so forth. This iterative process will be terminated if a *Nash-stable* partition, where no agent can benefit by its unilateral deviation, is determined.

**Definition 1** (*Nash-stable partition*)**.** Given an instance $(\mathcal{A}, \mathcal{T})$, a partition $\Pi$ is said to be *Nash-stable* if it holds that $e_i(S_{\Pi(i)}) \leq e_i(S_j)$ $\forall S_j \in \Pi$ for every agent $a_i \in \mathcal{A}$.

Our proposed approach will use an algorithm for *the 0/1 minimisation knapsack problem* (MinKP, for short) [12] as its subroutine. MinKP is defined as:

**Definition 2** (*the 0/1 minimisation knapsack problem*)**.** Suppose that there are a knapsack with its minimum requirement $R$ and a set of $n$ items $Z = \{z_1, ..., z_n\}$, where each item $z_i$ has its value $v_i$ and cost $c_i$. The objective is to pack the knapsack with the items so that the total value of all inserted items exceeds the minimum requirement while minimising the resultant total cost:

$$\min_{\{y_i \in \{0,1\}\}} \quad \sum_{i=1}^{n} c_i y_i \qquad \text{s.t.} \quad \sum_{i=1}^{n} v_i y_i \geq R$$

where $y_i = 1$ if item $z_i$ is inserted in the knapsack. Let $MinKP(Z, R, \mathcal{V}, \mathcal{C})$ denote an algorithm for MinKP, where $\mathcal{V} = \{v_1, ..., v_n\}$ and $\mathcal{C} = \{c_1, ..., c_n\}$. The output of this algorithm is the set of selected item for the knapsack.

### B. Design of an Agent's Expense Function

The expense of agent $a_i$'s with regard to coalition $S_j$ is defined as follows:

$$e_i(S_j) = \begin{cases} \tilde{c}_{ij} & \text{if } \sum_{\forall a_k \in S_j \cup \{a_i\}} w_{kj} < R_j \\ \tilde{c}_{ij} & \text{else if } a_i \in \hat{S}_j \\ \infty & \text{otherwise} \end{cases} \tag{2}$$

where $\tilde{c}_{ij}$ is the *learnt cost* of agent $a_i$ for task $t_j$, and $\hat{S}_j$ is the set of agents eligible to join the coalition for task $t_j$. The details of $\tilde{c}_{ij}$ and $\hat{S}_j$ will be described in following paragraphs. Initially, we set $\tilde{c}_{ij}$ to $c_{ij}$, which will be called the *original cost* in the rest of this paper. As shown in Eqn (2), agent $a_i$ regards the expense $e_i(S_j)$ as the learnt cost $\tilde{c}_{ij}$

in either of the following two cases: (i) there is no *conflict* between agents in $S_j \cup \{a_i\}$; or (ii) agent $a_i$ is still *eligible* to join the coalition for task $t_j$ even if there is a conflict.

The existence of a conflict can be examined by (C2). If the inequality does not hold, there is no conflict. Otherwise, sufficient agents in the same coalition provide more work capacities than required, meaning that some of them may not be needed. This situation causes a conflict such that superfluous agents are to be appropriately chosen and expelled. In this case, agent $a_i$ can ascertain its eligibility to join the coalition by an algorithm for MinKP: if $a_i \in \hat{S}_j$, where $\hat{S}_j = MinKP(S_j \cup \{a_i\}, R_j, \mathcal{W}_j, \mathcal{C}_j)$[1], then it is eligible. If the agent does not belong to one of the above two cases, then its expense is set to infinity.

However, $\hat{S}_j$ may differ depending on the existing coalition $S_j$. For examples, $MinKP$ selects agent $a_1$ instead of $a_2$ in some cases, vice versa in the other cases. It was observed that this fact sometimes prevent the agents from converging to a Nash-stable partition.

This possible divergence can be addressed by utilising learnt costs instead of original costs. The learnt cost of agent $a_i$ with regard to task $t_j$ is updated, while the agent is learning, as:

$$\tilde{c}_{ij} \leftarrow \tilde{c}_{ij} + \lambda \cdot c_{ij} \tag{3}$$

where $\lambda > 0$ is the *learning rate*. Initially, $\tilde{c}_{ij} = c_{ij}$. Whenever agent $a_i$ changes its decision from task $t_j$ to another (or possibly void task), the agent *learns* that the previously chosen task is not suitable for itself and penalises the task gradually by the learning rate as Equation (3). This can be called as *tabu learning* [18]. This iterative decision-making and learning process develops the partition of agents as well as their learnt costs, by which conflicts between agents can be resolved and a Nash-stable partition can be determined (the proof is provided in Section IV-A). When the process terminates, the resultant objective function value is calculated as Equation (1) by using original costs $c_{ij}$ instead of expenses in Equation (2), which are auxiliary variables only to find a conflict-free assignment.

### C. Algorithm

For simplicity of description, we assume that the agents are fully-connected and can access to the shared memory, where the current partition $\Pi[\tau]$ is stored. At each iteration $\tau$, a single agent exclusively updates the partition $\Pi[\tau]$ if necessary. However, it is worth noting that this assumption can be relaxed to a strongly-connected communication network (please refer to Remark 1).

The procedure of agent $a_i$ at iteration $\tau$ is described in Algorithm 1. Given the partition $\Pi[\tau]$, agent $a_i$ investigates every task $t_j$ according to Equation (2), and stores the corresponding expense as $E_j$ (Lines 1–15). If the least expense value provides infinity, then the agent joins $S_0$ (Lines 18–19). Else if it is lower than the existing one, the agent joins the newly-found coalition $S_{j^*}$ (Lines 20–21). If the agent deviates the existing coalition, it updates the learnt cost for the previously-chosen task $t_{\Pi[\tau](i)}$ and amends the current partition $\Pi[\tau+1]$ (Lines 23–24) to reflect its new decision. Note that the agent shares only $\Pi[\tau+1]$ with other agents, keeping the learnt costs locally.

[1]Please refer to Definition 2. Here, $\mathcal{W}_j = \{w_{kj} \mid \forall a_k \in S_j \cup \{a_i\}\}$ and $\mathcal{C}_j = \{c_{kj} \mid \forall a_k \in S_j\} \cup \{\tilde{c}_{ij}\}$.

**Algorithm 1** Decision-making of agent $a_i$ at iteration $\tau$

*Preknown:* $\lambda$, $\{c_{kj}\}$, $\{w_{kj}\}$, $\{R_j\}$ $\forall k$ $\forall j$
*Input:* $\Pi[\tau]$, $\{\tilde{c}_{ij}\}$ $\forall j$
*Output:* $\Pi[\tau+1]$, $\{\tilde{c}_{ij}\}$ $\forall j$
1: **for** each task $t_j \in \mathcal{T}$ **do**
2: Obtain the existing coalition info $S_j$ from $\Pi[\tau]$
3: **if** $R_j > \sum_{\forall a_k \in S_j \cup \{a_i\}} w_{kj}$ **then** // *No conflict*
4: $E_j = \tilde{c}_{ij}$;
5: **else** // *If there is a possible conflict*
6: $\mathcal{W}_j = \{w_{kj} \mid \forall a_k \in S_j \cup \{a_i\}\}$;
7: $\mathcal{C}_j = \{c_{kj} \mid \forall a_k \in S_j\} \cup \{\tilde{c}_{ij}\}$;
8: $\hat{S}_j = MinKP(S_j \cup \{a_i\}, R_j, \mathcal{W}_j, \mathcal{C}_j)$;
9: **if** $a_i \in \hat{S}_j$ **then**
10: $E_j = \tilde{c}_{ij}$;
11: **else**
12: $E_j = \infty$;
13: **end if**
14: **end if**
15: **end for**
16: $j^* = \text{argmin}_{\forall j} E_j$;
17: **if** $j^* \neq \Pi[\tau](i)$ **then** // *The best is not the current one*
18: **if** $E_{j^*} = \infty$ **then** // *No coalition is preferred*
19: Join $S_0$;
20: **else**
21: Join $S_{j^*}$;
22: **end if**
23: $\tilde{c}_{i,\Pi[\tau](i)} = \tilde{c}_{i,\Pi[\tau](i)} + \lambda \cdot c_{i,\Pi[\tau](i)}$;
24: Update and broadcast $\Pi[\tau+1]$;
25: **end if**

When updating $\Pi[\tau+1]$ (Line 24), it would also be possible to amend statuses of other agents who will be expelled by agent $a_i$ (i.e., $\forall a \in S_{j^*} \setminus \hat{S}_{j^*}$). However, this idea may increase the communication transactions at every iteration. To avoid this, Algorithm 1 is designed such that agent $a_i$ only notifies its decision change to others. Although this might temporarily mislead some agents to assume that they are still eligible for the existing coalitions, they can eventually notice their ineligibilities (please refer to Section IV-A) if the MinKP algorithm in Line 8 holds the following condition:

**Condition 1.** Suppose that $\hat{S} = MinKP(S, ...)$. The algorithm holds that $\hat{S} = MinKP(\hat{S} \cup C, ...)$, $\forall C \subseteq S \setminus \hat{S}$.

This paper utilises the approximation algorithm "Min-Greedy" in [12], which holds this condition.

## IV. Analysis

### A. Convergence to a Nash-stable Partition

Firstly, we show that it is enough for agent $a_i$ in Algorithm 1 to notify only its decision change to others. Suppose that

agent $a_i$ decides to join the coalition for task $t_j$ at iteration $\tau$, and thereby $a_l$ must be expelled from the coalition (i.e., $a_l \in S_j \setminus \hat{S}_j$). At the next iteration, in the case that agent $a_l$ executes $MinKP$ holding Condition 1, the agent realises that it is not supposed to be in the existing coalition, and leaves from there.

In the other case, another arbitrary agent $a_m$ misconceiving that agent $a_l$ is still in $S_j$ executes Algorithm 1, and may also join the coalition for task $t_j$. When agent $a_m$ obtains $\hat{S}_j = MinKP(S_j \cup \{a_i\} \cup \{a_m\}, ...)$, $\hat{S}_j$ may include $a_l$ because the combination of $a_l$ and $a_m$ may be preferred than some existing agent (i.e., the agents' eligibilities for the coalition may be changed depending on the input agent set). As an extreme case, suppose that all the other agents consequently join the coalition without expelling any of existing agents in this way, and the lastly joined agent obtains $\hat{S}_j^\star = MinKP(\mathcal{A}, ...)$. From the next iteration, agents who are supposed to be expelled (i.e., $\forall a \in \mathcal{A} \setminus \hat{S}_j^\star$) finally notice their ineligibilities for task $t_j$ because $MinKP$ holds Condition 1.

**Theorem 1.** *Given an instance $(\mathcal{A}, \mathcal{T})$, Algorithm 1 terminates and converges to a Nash-stable partition within a finite number of iterations.*

*Proof.* Suppose the statement is false, and there is an agent $a_\infty$ who infinitely changes its decision amongst a set of task $\mathcal{T}_\infty \subseteq \mathcal{T}$, preventing convergence to a Nash-stable partition. Whenever the agent revises its decision from a task, the learnt cost for the task is increased by the tabu learning (Line 23). In the case that $\mathcal{T}_\infty \subset \mathcal{T}$, all the learnt costs for tasks in $\mathcal{T}_\infty$ finally become more expensive than those for any tasks in $\mathcal{T} \setminus \mathcal{T}_\infty$ as the agent travels within $\mathcal{T}_\infty$. Thus, the agent will eventually join $S_j$ for any task $t_j \in \mathcal{T} \setminus \mathcal{T}_\infty$, which contradicts our supposition.

If $\mathcal{T}_\infty = \mathcal{T}$, agent $a_\infty$ finally cannot select any task in $\mathcal{T}$ and then join $S_0$ (Lines 18–19), because the agent rules out itself due to the increased learnt costs and the expense function in Equation (2). This fact also contradicts our supposition, which completes the proof. □

Since it is assumed that there are sufficient agents for a mission, there must be some agents in $S_0$. This implies that a Nash-stable partition provides a feasible assignment satisfying all the constraints (C1)–(C3).

### *B. Computation & Communication Complexities*

The computational complexity of Algorithm 1 for an agent at each iteration is $O(\sum_{\forall t_j \in \mathcal{T}} f(|S_j|+1))$, where $f(n)$ is the running time for $MinKP$ when $n$ items are given as inputs. This complexity is lower than just $O(f(|\mathcal{A}|))$ if $f(n)$ is super-additive (i.e., $f(n_1+n_2) \geq f(n_1)+f(n_2)$). This is the case for $MinKP$ used in this paper, i.e., $O(n\ln(n))$ [12]. Additionally, in practice, there may be some redundant agents who belong to $S_0$. This fact also makes the complexity of the proposed algorithm much less than $O(f(|\mathcal{A}|))$. Overall, it can be conservatively said that the convergence time of the proposed approach is $O(|\mathcal{T}| \cdot f(|S_{max}|) \cdot C)$. Here, $S_{max} = \mathrm{argmax}_{\forall \tau, \forall S_j \in \Pi[\tau] \setminus S_0} |S_j|$ is the maximum-size coalition until convergence except $S_0$. $C$ is the number of iterations until convergence, which will be experimentally investigated in Section V.

The required communicational traffic for an agent at each iteration is $O(|\mathcal{A}|-1)$, as the agent needs to notify only its status change to at most $|\mathcal{A}|-1$ of other agents even in a multi-hop fashion. Thus, the overall communicational complexity is $O(|\mathcal{A}| \cdot C)$.

**Remark 1** (Relaxation to a strongly-connected network)**.** By incorporating the stamp-based mutex subroutine in our previous work [19], the proposed algorithm can be asynchronously implemented even under a strongly-connected network of the agents. In this case, all the analysis described in Section IV are still valid. The number of required iterations $C$ may vary depending on the communication network. This effect will be investigated in the following section.

## V. Empirical Validation

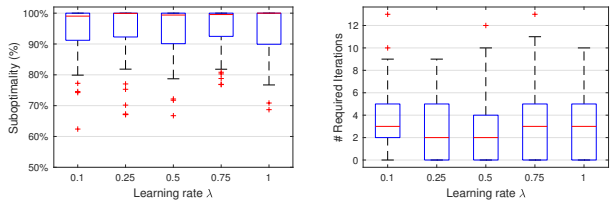
This study performs numerical experiments to validate the characteristics of the proposed approach regarding its suboptimality and the number of required iterations until converging to a Nash-stable partition. Note that we adopt the stamp-based mutex subroutine in [19] to examine the effect of a strongly-connected (SC) network or the fully-connected (FC) network.

For small-sized instances ($n_a = 10$, $n_t = 3$), we conduct 100 runs of Monte-Carlo experiments and investigate the proposed algorithm, compared with the optimal results obtained by a brute-force algorithm. For each instance, the parameters for agent $a_i$ and task $t_j$ are uniform-randomly generated from the following ranges: $R_j \in [5, 10]$; $c_{ij} \in [0.1, 1]$; and $w_{ij} \in [n_t/n_a, 0.9] \times R_j$. Moreover, in order to identify the effect of the learning rate, it is set as $\lambda \in \{0.1, 0.25, 0.5, 0.75, 1\}$. For each instance, a strongly-connected network is randomly generated.

For the rest of this experiment, we set an initial assignment as follows. Every agent firstly chooses the task that gives the least expense without consideration of other agents. Based on this partition, the agent checks its eligibility to stay the most preferred coalition, and goes to $S_0$ if it is not eligible. The resultant partition from this process is the initial assignment, and then an arbitrary agent begins to execute Algorithm 1.

Figure 1(a) shows that the mean value of the suboptimality is approximately $95\%$ and its standard deviation is less than $8\%$, regardless of the learning rate. In Figure 1(b), it is also presented that the number of required iterations until convergence is irrespective of the learning rate.

We also conduct Monte-Carlo experiments for large-sized instances where $(n_t, n_a) \in \{(5, 50), (5, 100)\}$. The problem-related parameters are drawn as follows: $c_{ij} \in [c_{min}\ 1]$; $R_j \in [5, 10]$; and $w_{ij} \in [n_t/n_a, w_{max}] \times R_j$. We also variously set $c_{min} \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$ and $w_{max} \in \{n_t/n_a, 0.2, 0.3, 0.5, 0.7, 0.9\}$. For every combinational case of $c_{min}$ and $w_{max}$, 100 instances are uniform-randomly generated from the aforementioned ranges. Here, we set $\lambda$




(a) Suboptimality (b) # Required Iterations

Fig. 1. Parametric analysis regarding the learning rate $\lambda$ ($n_a = 10, n_t = 3$) (a) the suboptimality (i.e., $f_{opt}/f_A$, where $f_{opt}$ is the total cost by a brute-force algorithm and $f_A$ is that by the proposed algorithm); (b) the number of iterations required for convergence to a Nash-stable partition.

to 0.25. We also perform the same experiments under the fully-connected communication network.

For such large-sized instances, since it is not feasible to obtain the optimal solution within a reasonable computation time, we instead utilise the *Linear-and-Conflict Relaxation* (LCR) solution to analyse the suboptimality of the proposed approach. The LCR solution is the outcome when (C1) and (C3) are relaxed. This can be simply determined as follows: fill every knapsack (i.e., task) with all the given items (i.e., agents) in a greedy manner until (C2) is satisfied no matter which items are already assigned to other knapsacks (i.e., conflict-relaxed); and the last inserted item can be included fractionally (i.e., linear-relaxed). It is obvious that $f_{LCR} \leq f_{opt}$, where $f_{LCR}$ denotes the objective function value of the LCR solution. From this, the suboptimality of the proposed algorithm can be conservatively investigated by comparing with $f_{LCR}$ instead of $f_{opt}$, because the true suboptimality (i.e., $f_{opt}/f_A$) is lower-bounded by $f_{LCR}/f_A$, which we call *the quasi-suboptimality*. This approach is inspired from [20], where greedy algorithms for the 0/1 single knapsack problem are investigated by comparing with the linear relaxation solution and the optimal solution.

Figure 2 shows the mean and the worst values regarding the quasi-suboptimality and the number of required iterations, amongst 100 instances at each combinational case of $w_{max}$ and $c_{min}$ under either the FC network or a SC network. Overall, the quasi-suboptimality is at least more than 50% at any of the cases, as shown in the-second-row subfigures. It is presented that as $w_{max}$ is decreasing and $c_{min}$ is increasing, the quasi-suboptimality becomes higher. One possible explanation is that agents in this setting become almost homogeneous with having much smaller $w_{ij}$ compared with $R_j$. This tendency makes the problem trivial so that an almost-optimal solution can be found by just assigning arbitrary agents to any tasks. The-third-row and the-fourth-row subfigures illustrate that the number of required iterations stays at a similar level of $n_a$ in the FC test cases, but increases in the SC test cases.

We further investigate the algorithmic complexity and the quasi-suboptimality with various $n_a$ and $n_t$. The results of 100 simulation runs at $c_{min} = 0.9$ and $w_{max} = 0.4$ are provided in Figure 3 and 4. In all the test cases, more iterations are required under SC than FC. The number of required iterations linearly increases with respect to $n_a$ or $n_t$ under FC, whereas, under SC, the algorithmic complexity is shown to be more affected by $n_t$. On the other hand, the quasi-suboptimality is slightly reduced by increasing $n_a$ or $n_t$, but still stays reasonable.

## VI. Conclusion

This paper studied the MRTA problem where each task has its minimum workload requirement to be fulfilled by multiple heterogeneous agents. The objective is to find an assignment that minimises the total cost of assigned agents while satisfying the tasks' requirements. We proposed a distributed game-theoretical approach, where each agent selfishly joins the most favourite coalition, and showed that a Nash-stable partition can always be determined with tabu-learning heuristics. Our experimental results revealed that the suboptimality of the proposed algorithm is at least more than 50%, and the number of required iterations until convergence remains the same order of the number of given agents at various problem settings.

More research is required to analytically evaluate the suboptimality of the proposed algorithm, especially connecting with the approximation ratio of its subroutine for MinKP. Furthermore, a natural progression of this work is to study how to exploit finally-unassigned agents depending on the context of a given mission. It would also be interesting to address a mixed mission scenario where there are additionally some robots capable of executing a task alone.

## References


[1] H.-S. Shin and P. Segui-Gasco, "UAV Swarms: Decision-Making Paradigms," in *Encyclopedia of Aerospace Engineering*. John Wiley & Sons, 2014, pp. 1–13.

[2] K. Barton and D. Kingston, "Systematic Surveillance for UAVs: A Feedforward Iterative Learning Control Approach," in *American Control Conference*, Washington, DC, USA, 2013, pp. 5917–5922.

[3] I. Bekmezci, O. K. Sahingoz, and S. Temel, "Flying Ad-Hoc Networks (FANETs): A Survey," *Ad Hoc Networks*, vol. 11, no. 3, pp. 1254–1270, 2013.

[4] I. Jang, J. Jeong, H.-S. Shin, S. Kim, A. Tsourdos, and J. Suk, "Cooperative Control for a Flight Array of UAVs and an Application in Radar Jamming (in press)," in *Proceedings of the IFAC World Congress*, Toulouse, France, 2017.

[5] B. P. Gerkey and M. J. Matarić, "A Formal Analysis and Taxonomy of Task Allocation in Multi-robot Systems," *International Journal of Robotics Research*, vol. 23, no. 9, pp. 939–954, 2004.

[6] T. Öncan, "A Survey of the Generalized Assignment Problem and Its Applications," *INFOR: Information Systems and Operational Research*, vol. 45, no. 3, pp. 123–141, 2007.

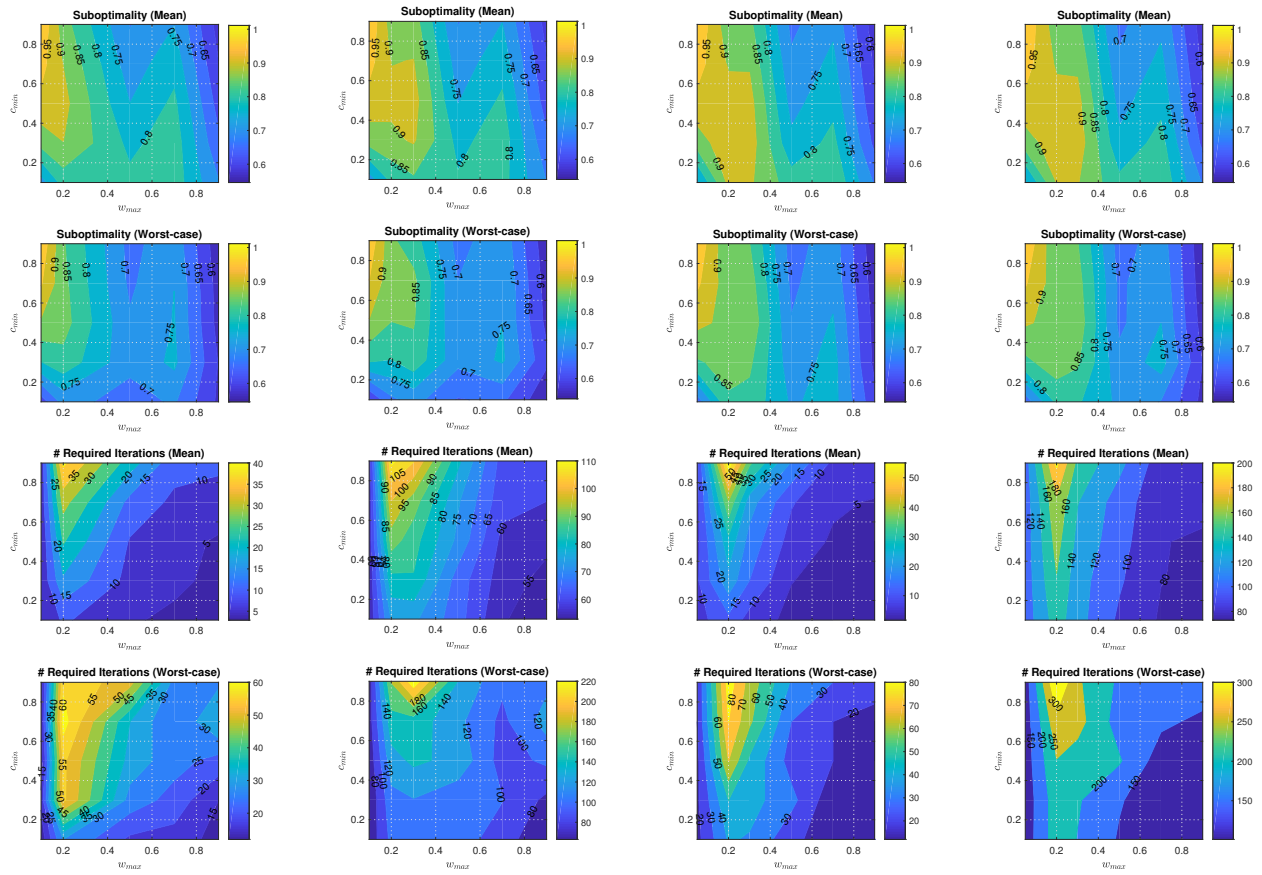
[7] T. William Mather and M. Ani Hsieh, "Distributed Robot Ensemble Control for Deployment to Multiple Sites," in *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, 2011.

[8] P. Segui-Gasco, H.-S. Shin, A. Tsourdos, and V. J. Segui, "Decentralised Submodular Multi-Robot Task Allocation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Hamburg, Germany, 2015, pp. 2829–2834.

[9] H. L. Choi, L. Brunet, and J. P. How, "Consensus-based Decentralized Auctions for Robust Task Allocation," *IEEE Transactions on Robotics*, vol. 25, no. 4, pp. 912–926, 2009.

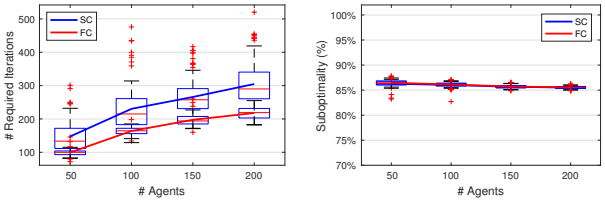
[10] L. Luo, N. Chakraborty, and K. Sycara, "Distributed Algorithms for Multirobot Task Assignment With Task Deadline Constraints," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 3, pp. 876–888, 2015.

[11] ——, "Provably-Good Distributed Algorithm for Constrained Multi-Robot Task Assignment for Grouped Tasks," *IEEE Transactions on Robotics*, vol. 31, no. 1, pp. 19–30, 2015.



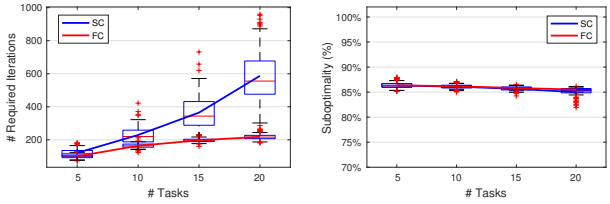

(a) $n_t = 5, n_a = 50$ (FC) (b) $n_t = 5, n_a = 50$ (SC) (c) $n_t = 5, n_a = 100$ (FC) (d) $n_t = 5, n_a = 100$ (SC)

Fig. 2. Contour plots of the quasi-suboptimality and the number of iterations required for convergence to a Nash-stable partition at every combinational case of $w_{max}$ and $c_{min}$ under either the fully-connected network (FC) or a strongly-connected network (SC). The-first-row and the-second-row subfigures show the mean and the minimum values of the quasi-suboptimality, and the-third-row and the-fourth-row subfigures show the average and the maximum values of the number of required iterations, respectively, amongst 100 uniform-randomly-generated instances at each case.




(a) # Required Iterations (b) Quasi-suboptimality

Fig. 3. Parametric analysis regarding $n_a$ ($n_t = 10$)




(a) # Required Iterations (b) Quasi-suboptimality

Fig. 4. Parametric analysis regarding $n_t$ ($n_a = 100$)


[12] M. M. Güntzer and D. Jungnickel, "Approximate Minimization Algorithms for the 0/1 Knapsack and Subset-Sum Problem," *Operations Research Letters*, vol. 26, no. 2, pp. 55–66, 2000.

[13] X. Han and K. Makino, "Online Minimization Knapsack Problem," in *Approximation and Online Algorithms. WAOA 2009. Lecture Notes in Computer Science*. Springer, Berlin, Heidelberg, 2010, vol. 5893, pp. 182–193.

[14] M. J. Geiger and W. Wenger, "On the Assignment of Students to Topics: A Variable Neighborhood Search Approach," *Socio-Economic Planning Sciences*, vol. 44, no. 1, pp. 25–34, 2010.

[15] M. Bender, C. Thielen, and S. Westphal, "A Constant Factor Approximation for the Generalized Assignment Problem with Minimum Quantities and Unit Size Items," in *Mathematical Foundations of Computer Science 2013. MFCS 2013. Lecture Notes in Computer Science, vol 8087*. Springer, Berlin, Heidelberg, 2013, pp. 135–145.

[16] S. O. Krumke and C. Thielen, "The Generalized Assignment Problem with Minimum Quantities," *European Journal of Operational Research*, vol. 228, no. 1, pp. 46–55, 2013.

[17] Z. Han, D. Niyato, W. Saad, T. Basar, and A. Hjørungnes, *Game Theory in Wireless and Communication Networks : Theory, Models, and Applications*. Cambridge University Press, 2012.

[18] D. Beyer and R. Ogier, "Tabu Learning: A Neural Network Search Method for Solving Nonconvex Optimization Problems," in *Proceedings of 1999 IEEE International Joint Conference on Neural Networks*. IEEE, 1991, pp. 953–961.

[19] I. Jang, H.-S. Shin, and A. Tsourdos, "Anonymous Hedonic Game for Task Allocation in a Large-scale Multiple Agent System," *IEEE Transactions on Robotics (Submitted)*, 2016.

[20] N. N. Galim'yanova, A. A. Korbut, and I. K. Sigal, "Ratios of Optimal Values of Objective Functions of the Knapsack Problem and Its Linear Relaxation," *Journal of Computer and Systems Sciences International*, vol. 48, no. 6, pp. 906–913, 2009.

2017-11-09

# Game-theoretical approach to heterogeneous multi-robot task assignment problem with minimum workload requirements

Jang, Inmo

IEEE

Inmo Jang, Hyo-Sang Shin and Antonios Tsourdos. Game-theoretical approach to heterogeneous multi-robot task assignment problem with minimum workload requirements. 2017 International Workshop on Research, Education and Development on Unmanned Aerial Systems (RED-UAS 2017), 3-5 October, Linkoping, Sweden.

https://dspace.lib.cranfield.ac.uk/handle/1826/12798