Robotics and Autonomous Systems 161 (2023) 104314

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Multi-robot task allocation clustering based on game theory

Javier G. Martin *, Francisco Javier Muros, José María Maestre, Eduardo F. Camacho

Department of Systems and Automation Engineering, University of Seville, Spain

ARTICLE INFO

Article history:
Received 3 February 2022
Received in revised form 1 October 2022
Accepted 14 November 2022
Available online 23 November 2022

Keywords:
Multi-robot systems (MRS)
Multi-robot task allocation (MRTA)
Clustering
Task planning
Cooperative game theory
Shapley value

ABSTRACT

A cooperative game theory framework is proposed to solve multi-robot task allocation (MRTA) problems. In particular, a cooperative game is built to assess the performance of sets of robots and tasks so that the Shapley value of the game can be used to compute their average marginal contribution. This fact allows us to partition the initial MRTA problem into a set of smaller and simpler MRTA subproblems, which are formed by ranking and clustering robots and tasks according to their Shapley value. A large-scale simulation case study illustrates the benefits of the proposed scheme, which is assessed using a genetic algorithm (GA) as a baseline method. The results show that the game theoretical approach outperforms GA both in performance and computation time for a range of problem instances.

1. Introduction

Multi-robot systems (MRS) perform *tasks* in a cooperative and efficient manner in many applications, e.g., inspection [1], aerial filming [2], surveillance [3], agriculture [4], mobile edge computing (MEC) [5], warehouses [6], and robotic sensor networks (RSN) [7]. To this end, MRS need to solve multi-robot task allocation (MRTA) problems [8,9] so that the available resources are employed in the most profitable way. MRTA problems can be arranged according to the taxonomy proposed in [10] and further developed in [11], including the inter-dependencies between the tasks. Following the above taxonomy the problems can be classified as:

- *Single/Multi-task robot* (ST/MT) problems, depending on whether the robots can perform at most one or several tasks simultaneously.
- *Single/Multi-robot task* (SR/MR) problems, considering the absence or existence of tasks that require more than one robot to be accomplished.
- *Instantaneous/Time-extended assignment* (IA/TA) problems, depending on whether the information available allows planning future allocations.
- *No/In-schedule/Cross-schedule/Complex dependency* (ND/ID/XD/CD) problems, taking into account the different possible types of dependency between simple tasks and the inter-schedule dependency between complex tasks.

MRTA problems have been traditionally implemented in a *centralized* manner at the expense of high computation costs. MRTA centralized approaches are usually classified into optimal assignment problem (OAP) algorithms [12], which focus on finding the *optimal solution* of a constrained problem [13], and metaheuristic algorithms, where methods such as genetic algorithms (GA) [14] and ant-colony optimization (ACO) [15] provide a *suboptimal solution* [16]. In contrast to centralized MRTA, *distributed* approaches divide the overall problem into pieces containing partial information, improving scalability at the expense of reducing performance. Distributed schemes typically focus on market-based algorithms [17] with different robots auctioning and bidding for the different tasks [18–22]. Finally, a possibility half-way between centralized and distributed approaches is that of *clustering* [23–25], which seeks a tradeoff between performance and computational burden. In this way, sets of tasks and robots are partitioned into loosely coupled *clusters* that can be managed efficiently in parallel. The problem of partitioning large MRTA schemes into several smaller, manageable and mutually exclusive pieces is as complex as well-known NP-hard problems as the capacitated clustering problem (CPP) [26,27], which consists of dividing a set of elements into clusters with limited capacity and maximum similarity within the clusters. Therefore, heuristic methods such as [28] become necessary. That being said, clustering is not new in MRTA problems and has indeed been applied to forming robot coalitions to perform complex tasks, e.g., in [29]. Also, in [30], a correlation clustering technique that enables similar robots to form coalitions is explored; and, in [31], an evolutionary algorithm is proposed for the case in which robots must participate in different clusters.

Note that clustering and cooperation fit naturally in the cooperative game theory framework, where intelligent rational

* Corresponding author.
E-mail addresses: jgarmar@us.es (J.G. Martin), franmuros@us.es (F.J. Muros), pepemaestre@us.es (J.M. Maestre), efcamacho@us.es (E.F. Camacho).

https://doi.org/10.1016/j.robot.2022.104314

decision-makers, known as *players*, are involved in cooperation situations or *games*. In this context, players can make sacrifices in terms of their own welfare to improve overall performance, and any group of players or cluster is free to make agreements and unify strategies [32]. Applications of cooperative game theory to engineering problems are usual, with contributions in the field of cooperation structures [33], communication networks [34–37], smart grids [38,39], MRS [40], and MRTA problems [41,42].

In this work, cooperative game theory is proposed to develop an algorithm for large ST-SR-TA-XD MRTA clustering problems. Our motivation stems out from the OCONTSOLAR Project, [1] which proposes the use of fleet of robots to build dynamic irradiance maps that can help increase the performance of large-scale thermal power plants [43,44]. In particular, spatially distributed irradiance information can be very useful whenever there are parts of the plant covered by clouds, which requires monitoring and tracking the attenuation on solar irradiance to avoid, among others, problems such as strong temperature gradients in the heat-transfer fluid. Keeping in mind the above application, we consider here the problem of assigning a set of measurement tasks, presumably generated by a higher layer of the control system for the set of available robots. Due to the combinatorial nature of the problem, it can become intractable from a computational viewpoint, especially for large-scale systems where many robots and tasks can be expected. To relieve this issue, we propose a clustering approach to group sets of robots and tasks, which become the players of a game. The proposed method is based on the ordering of players introduced by the Shapley value [45], the best-known solution concept in cooperative game theory, with applications in water systems [46], biology [47], finance [48], or power networks [49], among others. Certainly, the Shapley value has been applied to multi-agent systems to form coalitions due to its relationship with the *marginal contribution* of each agent [50]. Also, this value has been widely used in the literature to perform rankings, in works such as [51], where the value of the nodes in a social network is assessed to classify them according to their level of influence; [52], where carbon quotas are assigned to the different regions of China to reduce carbon emissions; [53], where wines are graded; or [54,55], where methods for ranking the links in control networks that may include constraints are proposed. Finally, a key feature in this context is that the Shapley value can be approximated in polynomial time using *randomized methods* [56–59]. In fact, there is a renewed interest in fast methods for computing the Shapley value [60–62] due to its use in large-scale machine learning applications [63–65]. In this way, and as it will be shown in the results section, our method can outperform other heuristics such as GA, even for hard timing constraints regarding the computation of the MRTA problem.

The rest of this work is organized as follows. The problem statement is introduced in Section 2. Some game theory concepts used in this work are presented in Section 3. In Section 4, the Shapley value-based clustering algorithm for MRTA problems is proposed. Likewise, in Section 5, a large-scale case study is introduced to assess the algorithm, which is also compared to a GA and a fully centralized approach. Finally, conclusions are given in Section 6.

2. Problem statement

We consider sets of *robots* $\mathcal{R} = \{1, 2, \ldots, R\}$ and *tasks* $\mathcal{T} = \{1, 2, \ldots, T\}$. For convenience, let us define set $\mathcal{N} = \mathcal{R} \cup \mathcal{T}$. To solve the MRTA problem, the elements inside $\mathcal{R}$ and $\mathcal{T}$ need to be properly matched considering the following assumptions:

Assumption 1. We focus on ST-SR-TA MRTA problems [10,11], meaning (i) a robot can only perform one task at a time, (ii) tasks can be fulfilled by only one robot, and (iii) all the tasks that must be allocated to robots are known.

Assumption 2. For the sake of simplicity, robots are governed by a controller that reaches the assigned tasks positions, avoiding static and dynamic obstacles [66,67].

Under the premises above, let us introduce the MRTA clustering as

$$\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_{N_c}\}, \tag{1}$$

where $\mathcal{C}_h$, $h = 1, \ldots, N_c$ are *disjoint clusters* of robots and tasks verifying

$$\bigcup_h \mathcal{C}_h = \mathcal{N}, \quad \mathcal{C}_{h_a} \cap \mathcal{C}_{h_b} = \emptyset, \ \forall \mathcal{C}_{h_a}, \mathcal{C}_{h_b} \in \mathcal{C}, \mathcal{C}_{h_a} \neq \mathcal{C}_{h_b}. \tag{2}$$

Notice that the number of different clustering alternatives for $\mathcal{C}$ is upper bounded by the Bell Number [68]

$$B_N = \sum_{s=0}^{N} \left(\frac{1}{s!} \sum_{j=0}^{s} (-1)^{s-j} \binom{s}{j} j^N \right), \quad \text{with } N = |\mathcal{N}|. \tag{3}$$

In particular, clusters $\mathcal{C}_h$ composed only of robots or tasks are allowed in this definition. Nevertheless, as it will be shown later, they will be discouraged via penalties, specially those containing only tasks because task-only clusters lead to tasks not being performed as tasks can only be executed by robots. Hence, in clusters without robots, tasks would remain undone.

2.1. MRTA objective function

Centralized approaches for MRTA problems rely on functions that evaluate *allocations*, i.e., the plan to perform the tasks according to criteria such as the distance traveled by robots, their battery level and the time to fulfill the missions [14,69–71]. Following [71], the multi-criteria evaluation function used is

$$A = \arg\min_U \left(\sum_{t=1}^{T} \delta_t \eta_t(U) + \sum_{r=1}^{R} \lambda_r d_r(U) + \varphi(U) \right), \tag{4}$$

where $U = \left[u_{r,t}\right]_{r \in \mathcal{R}, t \in \mathcal{T}}$, with $u_{r,t}$ being nonnegative integer variables that indicate the order in which robot r performs task t. Note that $u_{r,t} = 0$ when robot r does not perform task t. Likewise, δ_t corresponds with the priority given to a certain task $t \in \mathcal{T}$, i.e., it should be larger for urgent tasks. Also, $\eta_t(U)$ measures the *completion time* necessary for a certain robot to accomplish task t in allocation U, which includes the time spent by this robot in completing this and all previous tasks in U, and also the robot displacements from the origin to the first task and between tasks.[2] Parameter λ_r refers to a certain penalty of using robot $r \in \mathcal{R}$, i.e., it should be smaller for robots that are preferred to be used, while $d_r(U)$ is the distance traveled by robot r in allocation U. Finally, $\varphi(U)$ strongly penalizes unfeasible allocations, e.g., if there is not enough power in the batteries to perform allocation U.

In a nutshell, it is not only the distance, but its *balance* with the travel duration, which needs to be considered in the allocation. In any case, the allocation provided by (4) does not guarantee an optimal assignment, i.e., it might be possible to find better allocations [71]. Note also that (4) highly depends on the relevance of each task, which may cause strong nonlinearities. This feature is

[1] Optimal Control of Thermal Solar Energy Systems. *H2020 ADG-ERC project (Grant Agreement 789051).* https://cordis.europa.eu/project/id/789051/es

[2] Note that the dependence of $\eta_t(U)$ on U makes the problem become XD MRTA [10,11].

suitable for problems with a heterogeneous robot fleet where the use of certain robots needs to be restricted so that they are used in the most important tasks. For instance, in [44,71,72] the fleet of robots is composed of both unmanned aerial vehicles (UAV), and unmanned ground vehicles (UGV). Since UAVs are faster, it is reasonable to allocate the most urgent or relevant tasks to them. On the other hand, UGVs, which have higher operational autonomy, will be assigned to perform the rest of the tasks.

2.2. Cluster evaluation

Since it may not be possible to solve (4) in real time for large problems, we propose to divide robots and tasks into *clusters* or *coalitions*, i.e., subsets of players, and then to apply (4) subsequently to find an allocation for each resulting cluster in a distributed fashion. To this end, we introduce a function $\boldsymbol{J}$ to associate a cost $J(\mathcal{S})$ to each coalition $\mathcal{S} \subseteq \mathcal{N}$ according to its robots and tasks, considering:

- The average distance between robots and tasks so that larger distances are penalized.
- The velocity of the robots in the cluster. The faster the robots, the better.
- The battery level of the robots in the cluster. Again, the higher it is, the better.
- The penalties of the robots and tasks in (4) must be taken into account so that neither critical tasks nor penalized robots are concentrated in a single cluster.
- The operation time to fulfill each single task, since the completion time $\eta_t(U)$ in (4) depends on these operation times and also on the travel times spent by the robots.
- Large cardinality of $\mathcal{S}$ must also be penalized.

According to all these premises, we define

$$J(\mathcal{S}) = \psi(\mathcal{S}) \left(\sum_{i \subseteq \mathcal{S}} J_i - \alpha_1 \hat{D}_{\mathrm{RT}} - \alpha_2 |\mathcal{S}| + \rho \right), \tag{5}$$

with

$$J_i = \begin{cases} \alpha_3 E_i + \alpha_4 v_i - \alpha_5 \lambda_i, & \text{if } i \in \mathcal{R}, \\ -(\alpha_6 \delta_i + \alpha_7 \tau_i), & \text{if } i \in \mathcal{T}, \end{cases} \tag{6}$$

and

$$\psi(\mathcal{S}) = \begin{cases} 0, & \text{if } \{\mathcal{S} \subseteq \mathcal{R}\} \vee \{\mathcal{S} \subseteq \mathcal{T}\} \vee \{\mathcal{S} \equiv \emptyset\}, \\ 1, & \text{otherwise.} \end{cases} \tag{7}$$

Several terms are considered by cost function (5). First, J_i is related to the cost of each element $i \in \mathcal{N}$, while $\hat{D}_{\mathrm{RT}}$ is the average distance from robots to tasks in coalition $\mathcal{S}$. Terms E_i and v_i refer respectively to the available energy and velocity of a robot, and τ_i defines the *operation time* that a specific task i needs to be accomplished. Parameters λ_i and δ_i, indicate respectively the penalty of using a robot and the priority of a task, similarly to their use in (4). Finally, term $\psi(\mathcal{S})$ filters the cases where there is only one type of player inside coalition $\mathcal{S}$, i.e., either robots or tasks, with $\psi(\mathcal{S}) = 0$ in those coalitions, and ρ is an arbitrarily large number that guarantees that $J(\mathcal{S})$ is positive. Notice that α_1 to α_7 are nonnegative weighting scalars with different effects over the resulting clustering. Using larger α_1 reduces the admissible distance between robots and tasks; similarly, increasing α_2 decreases the number of players in the formed clusters; also, increasing α_3, α_4, α_5, α_6, and α_7 prioritizes aspects such as the battery level, velocity, robot penalty, task relevance, and operation time, respectively. These weights can be set to 1 initially and then modified on the basis of the system designer's experience and their impact should be simulated in a trial and error fashion.

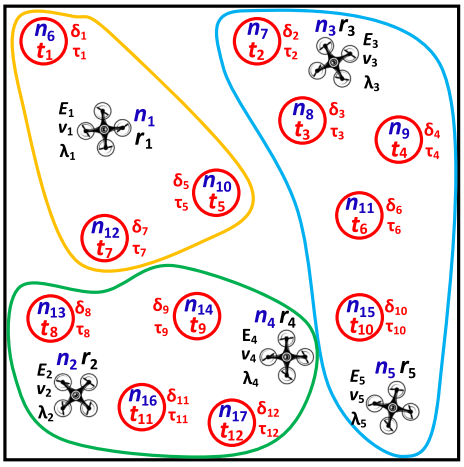

Fig. 1. Clustering academic example.

Remark 1. Unlike other works, we consider that tasks can also be players in the cooperative game, being characterized by their locations and other parameters such as their relevance and the operation time required to accomplish them. As can be extracted from (6), conversely to robots, tasks will have a negative effect on the cluster evaluation function (5).

Remark 2. The MRTA problem (4) has inspired the design of function (5), but alternative evaluation functions and different MRTA objective functions, such as that of [73], could be used as well.

Remark 3. Parameter α_2 in (5) penalizes the coalition size. Thus, lower/higher values of α_2 will favor larger/smaller coalitions. Note that the cluster size affects the allocation computation by (4). Indeed, larger clusters gain optimality at the expense of computation cost.

2.3. Clustering evaluation

Among all the available clustering options, whose amount is given by the Bell number (3), it is necessary to define a criterion to evaluate their performance. To this end, we define a function $V(\mathcal{C})$ to assess clustering $\mathcal{C}$ as

$$V(\mathcal{C}) = \Omega(\mathcal{C}) \sum_{\mathcal{C}_h \in \mathcal{C}} J(\mathcal{C}_h), \quad \text{s.t. (2)}, \tag{8}$$

with $J(\cdot)$ given by (5), and where $\Omega(\mathcal{C}) = 0$ if any $\mathcal{C}_h \in \mathcal{C}$ is composed only of tasks and is set to 1 otherwise. This function $V(\mathcal{C})$ can be maximized to obtain the optimal allocation.

A clustering method based on game theory is proposed in the following section. In this method, it is necessary to compute the value of all the possible coalitions $\mathcal{S} \subseteq \mathcal{N}$ by means of (5). Then, the cost for each specific clustering solution $\mathcal{C}$ (1) is evaluated by (8). Note that the cost of clusters $\mathcal{C}_h \in \mathcal{C}$ is computed by function J in (8), since these clusters belong to the full set of coalitions, i.e., $\mathcal{C}_h \subseteq \mathcal{N}$.

Example 1. Consider a set $\mathcal{N}$ composed of $R = 5$ robots and $T = 12$ tasks, i.e., 17 players, as shown in Fig. 1. The cost of the corresponding 2^{17} coalitions can be computed by (5). Then, from the $B_{17} \approx 82.86 \times 10^9$ options, a possible clustering $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3\}, \mathcal{C}_h \subseteq \mathcal{N}$ is represented, the cost of which would be computed by (8), with the parameters involved also shown

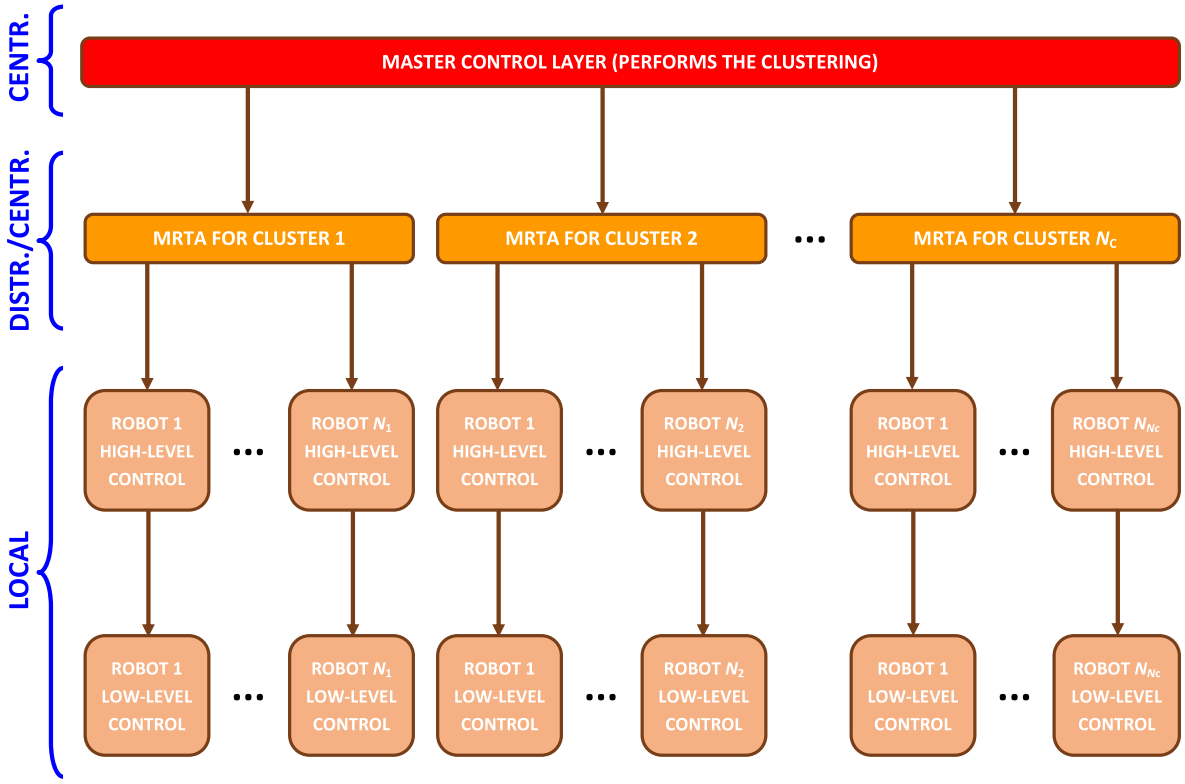

Fig. 2. Control scheme of the proposed approach. Note that the top layer is performed by a centralized controller while the bottom ones are local. The middle layer, which performs the MRTA problem for each cluster, can be either distributed or centralized. Notice that the number of elements in each cluster are respectively denoted by $N_1, N_2, \ldots, N_{N_c}$.

in Fig. 1. Focusing on cluster $\mathcal{C}_1 = \{r_1, t_1, t_5, t_7\}$, consider than the resulting allocation is given by $U_1 = [5, 7, 1]$. Then, for instance, note that $\eta_1(U_1) = \tau_5 + \tau_7 + \tau_1 + \frac{d_1(U_1)}{v_1}$.

2.4. Proposed control scheme

To conclude this section, the overall control scheme proposed in this work is illustrated in Fig. 2. On the top, a master control layer is in charge of performing the clustering, that is, dividing the robots and tasks into N_c different groups (recall (1)), to minimize function (8). Once the clustering is performed, a middle layer solves the MRTA problem for each cluster $\mathcal{C}_h, h = 1, \ldots, N_c$, providing a suitable allocation by establishing the sequence of tasks performed by each robot. Note that this allocation problem for each cluster could be solved either in a centralized [71,72,74–77] or a distributed [78–80] fashion. In this particular work, we consider the centralized solution introduced in [71], which is based on solving (4). Finally, once the allocation is sent to the robots, two different bottom layers are also locally required. The high-level layer plans the trajectory of each robot to perform its tasks and avoid obstacles. Also, a low-level layer manages technical aspects such as battery level, data buffer capacity, etc.

3. Game theory viewpoint

In this work, pair $(\mathcal{N}, \boldsymbol{J})$ will be interpreted as a cooperative game with transferable utility (TU-game) where $\mathcal{N}$ is the set of players and function $\boldsymbol{J}$ assigns cost $J(\mathcal{S})$ defined by (5) to each coalition $\mathcal{S} \subseteq \mathcal{N}$. Note that, to properly consider $\boldsymbol{J}$ as the characteristic function of a TU-game, a necessary condition is $J(\emptyset) = 0$, which is assured by $\psi(\mathcal{S})$ in (5).

From the different cooperative game theory payoff rules available in the literature, we will consider the Shapley value [45] to compute the relevance of the different coalitions among the different players – robots and tasks – involved in the game. The Shapley value assigns to game $(\mathcal{N}, \boldsymbol{J})$ the vector $\boldsymbol{\phi}(\mathcal{N}, \boldsymbol{J}) = \left[\phi_i(\mathcal{N}, \boldsymbol{J})\right]_{i \in \mathcal{N}}$, with

$$\phi_i(\mathcal{N}, \boldsymbol{J}) = \sum_{\mathcal{S} \subseteq \mathcal{N}: i \notin \mathcal{S}} \frac{S!(N - S - 1)!}{N!} [J(\mathcal{S} \cup \{i\}) - J(\mathcal{S})], \tag{9}$$

with $N = |\mathcal{N}|$, $S = |\mathcal{S}|$, and where $\gamma_i(\mathcal{S}, \boldsymbol{J}) = J(\mathcal{S} \cup \{i\}) - J(\mathcal{S})$ is the marginal contribution of player i when it incorporates to coalition $\mathcal{S}$. That is, the Shapley value can be interpreted as the expected marginal contribution for each player when it joins randomly to a coalition. Notice that $\frac{S!(N-S-1)!}{N!}$ is the probability for player i to join $\mathcal{S}$.

The key idea of our proposal is to use the ordering of players generated by the Shapley value to perform the clustering, as it has been previously done in other works for coalition formation, e.g., [50]. Even when this value is oriented to the distribution of payoffs among players in the grand coalition, the corresponding payoff is based on their average marginal contribution when they join a random coalition. Therefore, the Shapley value provides us with a measure of the relevance of every player that is weighted across the set of possible coalitions where it can participate, hence, as done in, e.g., [51–53], it can be used as a ranking criterion for players.

Notice that the Shapley value was introduced axiomatically as the only solution concept that satisfies the properties of *null player*, *symmetry*, *additivity* and *efficiency* [45]. In particular, the efficiency property states that the sum of all Shapley values remains *constant* by satisfying

$$\sum_{i \in \mathcal{N}} \phi_i(\mathcal{N}, \boldsymbol{J}) = J(\mathcal{N}), \tag{10}$$

which implies that the grand coalition is completely shared among the players, and will be of particular interest in the clustering algorithm presented later.

Alternatively to (9), the Shapley value can be rewritten in terms of all possible *orderings* of players in $\mathcal{N}$ coming into a coalition, i.e., $N!$. Hence, assuming equiprobable orderings, the Shapley value can be computed by [81]

$$\phi_i(\mathcal{N},\boldsymbol{J}) = \frac{1}{N!}\sum_{\pi\in\Pi(\mathcal{N})}\gamma_i^{\pi}(\mathcal{N},\boldsymbol{J}), \quad \forall i\in\mathcal{N}, \tag{11}$$

where

$$\begin{aligned}\gamma_i^{\pi}(\mathcal{N},\boldsymbol{J}) &= J(\{j\in\mathcal{N}\mid \pi(j)\le\pi(i)\})\\ &- J(\{j\in\mathcal{N}\mid \pi(j)<\pi(i)\}),\end{aligned} \tag{12}$$

is the marginal contribution of player i to the players that are ranked before it in permutation π, and with $\Pi(\mathcal{N})$ being the collection of all permutations.

Remark 4. Coalitions composed solely of either robots or tasks are *excluded* by means of term $\psi(\mathcal{S})$. Nevertheless, it is possible to use other methods in the line of [82,83] to exclude prohibited coalitions via a Shapley value redefinition.

3.1. Estimating the Shapley value

Computing the Shapley value by (9) becomes increasingly difficult due to the exponential growth of the problem size. Depending on the number of players involved in the problem, it may not be feasible to calculate $J(\mathcal{S})$ for all $\mathcal{S}\subseteq\mathcal{N}$. To solve this issue, there are several proposals in the literature [56–59] to estimate the Shapley value in polynomial time. In particular, the randomized algorithm proposed in [56] and improved in [57] is used in this work. Thus, starting from the formulation in (11), a set $\mathcal{Q}$ containing a sample of q different permutations π, taken with replacement and with equal probability from set $\Pi(\mathcal{N})$, is considered. Then, the Shapley value of each player is estimated by the average of the marginal contributions over set $\mathcal{Q}$, obtaining vector $\widetilde{\boldsymbol{\phi}}(\mathcal{N},\boldsymbol{J})$, which is defined by

$$\widetilde{\phi}_i(\mathcal{N},\boldsymbol{J}) = \frac{1}{q}\sum_{\pi\in\mathcal{Q}}\gamma_i^{\pi}(\mathcal{N},\boldsymbol{J}), \quad \forall i\in\mathcal{N}. \tag{13}$$

Expression (13) provides an estimation of the Shapley value with desirable properties such as *efficiency*. Furthermore, following the central limit theorem, it holds that the estimator follows a normal distribution characterized by [56]:

$$\widetilde{\phi}_i(\mathcal{N},\boldsymbol{J}) \sim \mathtt{N}\left(\phi_i, \frac{\sigma_{\phi_i}^2}{q}\right), \tag{14}$$

with

$$\sigma_{\phi_i}^2 = \frac{1}{N!}\sum_{\pi\in\Pi(\mathcal{N})}\left(\gamma_i^{\pi}(\mathcal{N},\boldsymbol{J}) - \phi_i(\mathcal{N},\boldsymbol{J})\right)^2, \quad \forall i\in\mathcal{N}. \tag{15}$$

Consequently, if the number of permutations q is chosen satisfying the following condition, $\forall i\in\mathcal{N}$:

$$q \ge \frac{Z_{\theta/2}^2\sigma_{\phi_i}^2}{\varepsilon^2}, \tag{16}$$

the estimation error is guaranteed to be bounded by

$$P\left(|\widetilde{\phi}_i(\mathcal{N},\boldsymbol{J}) - \phi_i(\mathcal{N},\boldsymbol{J})| \le \varepsilon\right) \ge 1-\theta, \quad \forall i\in\mathcal{N}, \tag{17}$$

with ε being the approximation error, $Z\sim\mathtt{N}(0,1)$, and where $Z_{\theta/2}^2$ is the value such that $P(Z\ge Z_{\theta/2}^2)=\theta/2$, with $0\le\theta\le 1$. Given that $\sigma_{\phi_i}^2$ is *a priori* unknown, in this work we assume $\varepsilon = \max_i \varepsilon_i$, with $\varepsilon_i = \xi\sigma_{\phi_i}$ and $\xi = \frac{1}{\sqrt{\beta}}$. Hence, the condition given by (16) is reduced to

$$q \ge \beta Z_{\theta/2}^2. \tag{18}$$

Given that we are interested in qualitative information about the different Shapley values regarding their relative position in a ranking, we propose here an iterative method starting with the solution provided in (18), i.e., $q_{\text{ini}} = \lceil \beta Z_{\theta/2}^2 \rceil$. Once this initial sample has been taken, $\widetilde{\phi}$ will be dynamically computed and the agents ranked by their estimated Shapley value. Then, more samples can be added to update $\widetilde{\phi}$ until the rank stays constant for a given number of steps $l_{\max}$ without variations. The proposed procedure for computing q is given in Algorithm 1, with l being a counter variable for the steps without ranking changes.

Algorithm 1: Procedure for estimating the Shapley value with a dynamic q criterion

```
Set q = q_ini and l = 0;
Compute φ̃ by (13);
Find ordering O with players ranked by their estimated
  Shapley values φ̃_i;
while l < l_max do
    q = q + 1;
    Recompute φ̃ by (13);
    Find ordering O' with players ranked by φ̃_i;
    if O = O' then
        l = l + 1;
    else
        l = 0;
    end
end
```

Remark 5. According to [56], $\widetilde{\phi}_i$ converges to ϕ_i when $q\to\infty$. Therefore, Algorithm 1 eventually provides us with a *stable* ranking based on the Shapley value with a certain error ε included in the specifications.

4. Clustering algorithm

The Shapley value ranks robots and tasks according to their expected marginal contributions. It can be used to organize clusters in order to enhance performance while *balancing* the computational burden. In particular, we propose to balance the clusters according to the sum of the Shapley values of their elements. To this end, each robot is initially set as the *leader* of a cluster composed only of itself. Tasks are then distributed leveling the *aggregate* Shapley values of the clusters, and a clustering result is obtained. It is remarkable that robots with higher Shapley values have the most to offer in terms of performance. On the other hand, the most demanding tasks are those with lower Shapley values. Therefore, we assign these tasks to the most capable robots. At this point, the two robots with the lowest Shapley values are merged in a new single player, the tasks are again distributed and a new clustering result is obtained. This process is recursively repeated by merging robots as many times as rounds of tasks distribution have been done, until the grand coalition of robots is achieved. Finally, from the set of possible clusterings computed in each round, the most appropriate one according to (8) is chosen as the solution. This method will be named *Shapley value clustering algorithm* (SVCA) and is detailed in Algorithm 2.

Remark 6. Robots and tasks typically have opposite signs regarding their Shapley values. Hence, grouping them into a single

Algorithm 2: Shapley value clustering algorithm (SVCA)

Let $\mathcal{N} = \mathcal{R} \cup \mathcal{T}$;
Set $s = 1$ and $q = q_{\text{ini}}$;
Compute $\widetilde{\phi}_i(\mathcal{N}, \boldsymbol{J})$ using (13) and Algorithm 1;
Start with $N_c = R$ clusters containing 1 robot;
while $N_c \geq 1$ **do**
  while $T > 0$ **do**
    Add task $i \in \mathcal{T}$ with the lowest $|\widetilde{\phi}_i(\mathcal{N}, \boldsymbol{J})|$ to the cluster containing robot $j \in \mathcal{R}$ with the highest $|\widetilde{\phi}_j(\mathcal{N}, \boldsymbol{J})|$. A new mixed robot-task player $a = \{i\} \cup \{j\}$ whose estimated Shapley value is $\widetilde{\phi}_i(\mathcal{N}, \boldsymbol{J}) + \widetilde{\phi}_j(\mathcal{N}, \boldsymbol{J})$ is created, and old players i and j are deleted;
  end
  Take the clusters created as a possible solution $\mathcal{C}^s$;
  $s = s + 1$;
  Reset original sets $\mathcal{R}$ and $\mathcal{T}$;
  $N_c = N_c - 1$;
  while $R > N_c$ **do**
    Take player $i \in \mathcal{R}$ with the lowest $|\widetilde{\phi}_i(\mathcal{N}, \boldsymbol{J})|$ and add it to player $j \in \mathcal{R}$ with the second lowest $|\widetilde{\phi}_j(\mathcal{N}, \boldsymbol{J})|$. Therefore, a new robot-robot player $b = \{i\} \cup \{j\}$ whose estimated Shapley value is $\widetilde{\phi}_i(\mathcal{N}, \boldsymbol{J}) + \widetilde{\phi}_j(\mathcal{N}, \boldsymbol{J})$ is created, and old players i and j are removed;
  end
end
Compute $V(\mathcal{C}^s)$ for all possible clusterings $\mathcal{C}^s$, with s $\in [1, N_c]$, by means of (8), and select the highest result as the clustering solution;

cluster balances the Shapley value of the merger. Consequently, robots with lower Shapley values tend to be grouped to form competitive clusters. Finally, note that all clusters initially contain one robot and therefore, the proposed algorithm prevents the formation of task-only clusters. Clusters composed only of robots are allowed because there is no need to assign all the robots. However, all the tasks must be assigned.

Remark 7. Algorithm 2 is suboptimal because it does not explore all possible coalition structures. Indeed, a full search may not be computationally affordable as the problem grows exponentially with the number of players by $O(N^N)$.

5. Case study

In this section, the proposed SVCA is applied to a simulated robot fleet that performs maintenance labors and data acquisition tasks in an industrial plant. In the context of the aforementioned OCONTSOLAR Project, the tasks will include the gathering of spatially distributed irradiance measurements throughout the plant and the maintenance of the solar collectors. For simplicity, it is assumed that robots can move straight to tasks without considering obstacles. Also, it has been considered that all robots can perform all tasks and are equipped with the same instrumentation. Hence, the time to perform a task only depends on the task. All the simulations in this section have been performed using Matlab® in a 3.2 GHz Intel® Core™ i7/16 GB RAM computer.

Random problems for a certain size N are considered here, assuming $R < T$, and where the locations of robots and tasks are determined randomly within a square map of 500 × 500 m. Weighting parameters are given by $\alpha_1 = 0.1$ (to allow large distances between robots and tasks), $\alpha_2 = 10$ (to limit the size of the clusters), and the rest of them, α_3 to α_7, neutral and equal to 1 (to show a cluster forming scenario based on robot and task parameters where the distances between players in a cluster are not very relevant). The remaining operating parameters of the players affecting (5) will also be randomly determined within the following ranges: $E \in [0, 100]$, $v \in [1, 20]$, $\lambda \in [1, 5]$, $\tau \in [0, 10]$, $\delta \in [1, 5]$, with the specific values shown in Table 1. According to these specifications, the scenario depicted in Fig. 3 has been randomly generated with $R = 15$ robots and $T = 45$ tasks.

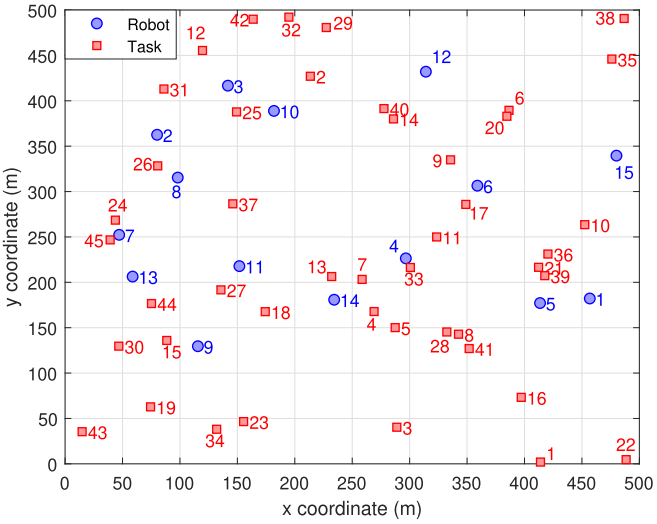

Fig. 3. Randomly generated scenario ($N = 60$).

Table 1
Randomly generated scenario parameters for robots and tasks ($r := i \in \mathcal{R}$ and $t := i \in \mathcal{T}$).

Robot	E_r	v_r	λ_r	Task	τ_t	δ_t	Task	τ_t	δ_t	Task	τ_t	δ_t
r_1	97	6	5	t_1	8	3	t_{16}	5	1	t_{31}	4	2
r_2	99	8	1	t_2	1	5	t_{17}	9	5	t_{32}	8	2
r_3	43	18	1	t_3	8	4	t_{18}	5	1	t_{33}	10	4
r_4	90	11	4	t_4	3	3	t_{19}	6	4	t_{34}	4	3
r_5	2	8	3	t_5	9	2	t_{20}	5	1	t_{35}	3	4
r_6	1	18	1	t_6	8	4	t_{21}	6	2	t_{36}	7	5
r_7	6	20	2	t_7	5	3	t_{22}	9	4	t_{37}	10	3
r_8	58	8	4	t_8	6	3	t_{23}	2	2	t_{38}	3	3
r_9	8	13	5	t_9	6	3	t_{24}	9	2	t_{39}	5	1
r_{10}	80	8	3	t_{10}	2	2	t_{25}	1	3	t_{40}	2	1
r_{11}	70	5	5	t_{11}	8	4	t_{26}	1	5	t_{41}	10	1
r_{12}	86	8	1	t_{12}	5	3	t_{27}	6	5	t_{42}	8	5
r_{13}	31	10	4	t_{13}	3	5	t_{28}	3	1	t_{43}	6	4
r_{14}	58	6	5	t_{14}	7	3	t_{29}	8	2	t_{44}	8	3
r_{15}	62	5	3	t_{15}	9	4	t_{30}	2	4	t_{45}	6	1

Henceforth, we will consider two different computation times to analyze the burden of our algorithm. In particular, t_{clu} is defined as the time required to solve the clustering and t_{all} refers to the time required to *sequentially* solve the allocation in all its clusters. Note that the allocation in the clusters could be solved in parallel, but we have considered a sequential process to perform a fair comparison with other algorithms in the literature.

5.1. Centralized clustering

At this point, it is possible to provide the centralized allocation of the aforementioned problem using (4), which is shown in Fig. 4 and results in an allocation cost $A = 61283.27$. Nevertheless, the computational burden of this solution is $t_{\text{all}} = 733.95$ s, which makes the centralized allocation unaffordable for large-scale problems.

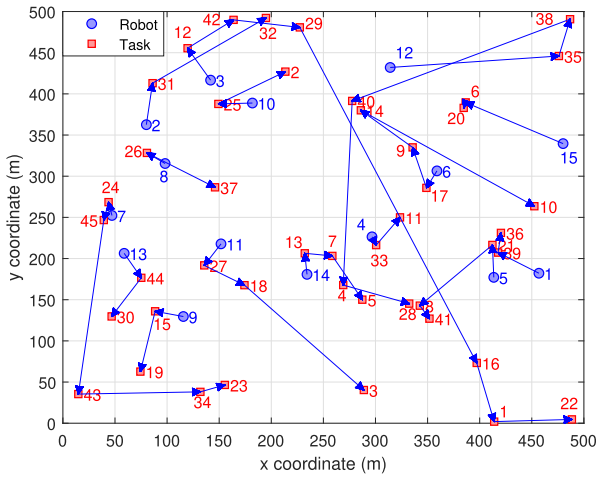

Fig. 4. Scenario solved via a centralized allocation (one cluster).

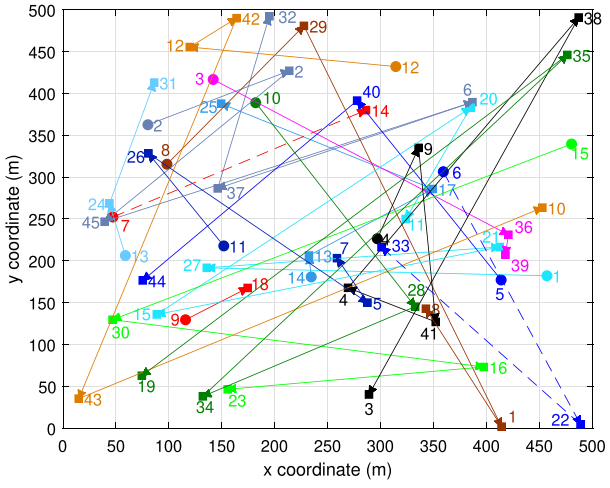

Fig. 5. Scenario clustered by the SVCA. Different colors represent the 13 different clusters while different line styles refer to different robots trajectories in the same cluster. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

5.2. SVCA clustering

The proposed SVCA ranks players according to their Shapley value. Given the large number of players ($N = 60$) we need to estimate the Shapley value taking $\theta = 0.1$, $\xi = 0.08$ and, thus, $\beta = 156$ in (18), which results in $q_{\text{ini}} = 423$. Using a stopping criterion of $l_{\max} = 10$ in Algorithm 1, we estimate the Shapley value after $q = 453$ permutations. Finally, Algorithm 2 is implemented, obtaining the clustering results of Fig. 5. The resulting clusters are detailed in Table 3, while the specific distance traveled by the robots and the tasks completion time due to the clustering solution is shown in Table 4. The clustering performance is given by $V(\mathcal{C}) = 10921.20$, while the allocation cost computed by (4) is $A = 83392.52$. Likewise, the overall time to perform the clustering and achieve the allocation is $t_{\text{clu}} + t_{\text{all}} = 8.53 + 1.57 = 10.10$ s. Note that the computation time is significantly reduced with respect to the centralized solution at the cost of a performance loss.

5.3. Genetic algorithms (GA)-based clustering

Since problem (8) is highly nonlinear, it is not possible to find the optimal clustering without evaluating all the possibilities,

Table 2
Genome involved in the GA. Note that the number of genes generated increases fast because the number of variations with repetitions is given by N^N.

Player	1	⋯	i	⋯	N
Cluster	$[1, N]$	⋯	$[1, N]$	⋯	$[1, N]$

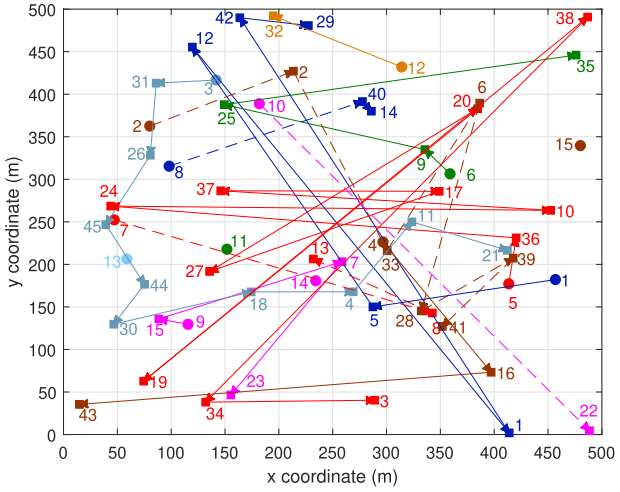

Fig. 6. Scenario clustered by the GA. As done in the previous case, different colors represent the eight different clusters and different line styles refer to different robots trajectories in the same cluster. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

which may not be computationally feasible. For this reason, in order to measure the performance of the proposed algorithm, we have considered the well-known *genetic algorithm* (GA) [84–86]. The rationale of GA is to start from a random population of *genes* representing *possible* solutions and improve them by mixing and mutating them. The *gen* used in GA has been defined as illustrated in Table 2, and it can be expressed mathematically as a vector $\boldsymbol{g} = [g_i]_{i\in\mathcal{N}}$, where $g_i \in [1, N]$ is the cluster containing player i. Finally, it is interesting to introduce the *population size*, i.e., the amount of genes that the algorithm uses in each generation. Indeed, increasing this parameter can improve the optimality of the solution achieved, but it also rises the overall computation time.

In this work, the GA has been implemented using Matlab® function ga with default settings. Specifically, the same scenario as before with $N = 60$ has been solved using GA. The clustering results are shown in Fig. 6, with the clusters formed and the clustering information being respectively shown in Tables 3 and 4, to ease the comparison with those of the SVCA. In this case, the clustering performance is $V(\mathcal{C}) = 5817.27$ and the allocation cost (4) is $A = 86399.29$. Also, the overall computation time is given by $t_{\text{clu}} + t_{\text{all}} = 9.05 + 1.83 = 10.88$ s. Note that the SVCA outperforms the GA in these results.

5.4. Averaging results

To reinforce our assessment, sets of 50 random problems solved 10 times to average the results have been computed. Indeed, random problems allow a proper study of the performance of the heuristic algorithms considered. First, the problem size is fixed to $N = 60$ and the rest of parameters for robots and tasks are chosen randomly. For SVCA, the Shapley value is estimated with $q_{\text{ini}} = 423$ as before ($\theta = 0.1$, $\xi = 0.08$). Note that the average computational load to solve the problem without clustering, i.e., in a centralized manner, is in the range

Table 3
Clusters obtained as a result of the application of SVCA and GA. It can be seen that the number of clusters of the proposed approach is higher than those of the genetic scheme, which in turn implies that clusters of GA contain more players and hence are more difficult to manage.

	SVCA	GA
$\mathcal{C}_1$	$\{r_1, t_{11}, t_{15}, t_{20}, t_{21}, t_{27}\}$	$\{r_1, r_8, t_1, t_5, t_{12}, t_{14}, t_{29}, t_{40}, t_{42}\}$
$\mathcal{C}_2$	$\{r_2, t_2, t_6, t_{32}, t_{37}, t_{45}\}$	$\{r_2, r_4, r_{15}, t_2, t_6, t_{16}, t_{28}, t_{33}, t_{39}, t_{41}, t_{43}\}$
$\mathcal{C}_3$	$\{r_3, t_{36}, t_{39}\}$	$\{r_3, t_4, t_{11}, t_{18}, t_{21}, t_{26}, t_{30}, t_{31}, t_{44}, t_{45}\}$
$\mathcal{C}_4$	$\{r_4, t_3, t_4, t_9, t_{38}, t_{41}\}$	$\{r_5, r_7, t_3, t_8, t_{10}, t_{13}, t_{17}, t_{19}, t_{20}, t_{24}, t_{27}, t_{34}, t_{36}, t_{37}, t_{38}\}$
$\mathcal{C}_5$	$\{r_5, r_6, t_{22}, t_{33}, t_{40}, t_{44}\}$	$\{r_6, r_{11}, t_9, t_{25}, t_{35}\}$
$\mathcal{C}_6$	$\{r_7, r_9, t_{14}, t_{18}\}$	$\{r_9, r_{10}, r_{14}, t_7, t_{15}, t_{22}, t_{23}\}$
$\mathcal{C}_7$	$\{r_8, t_1, t_8, t_{29}\}$	$\{r_{12}, t_{32}\}$
$\mathcal{C}_8$	$\{r_{10}, t_{19}, t_{28}, t_{34}, t_{35}\}$	$\{r_{13}\}$
$\mathcal{C}_9$	$\{r_{11}, t_5, t_7, t_{26}\}$	–
$\mathcal{C}_{10}$	$\{r_{12}, t_{10}, t_{12}, t_{42}, t_{43}\}$	–
$\mathcal{C}_{11}$	$\{r_{13}, t_{24}, t_{31}\}$	–
$\mathcal{C}_{12}$	$\{r_{14}, t_{13}, t_{17}, t_{25}\}$	–
$\mathcal{C}_{13}$	$\{r_{15}, t_{16}, t_{23}, t_{30}\}$	–

Table 4
Distance traveled by robots (in m) and completion time of tasks (in s) after solving the MRTA problem with both SVCA and GA clustering algorithms ($r := i \in \mathcal{R}$ and $t := i \in \mathcal{T}$). The best values have been highlighted in bold. Note that they must be multiplied by parameters λ_r and δ_t in Table 1 to compute the cost function.

Robot	d_r^{SVCA}	d_r^{GA}	Task	η_t^{SVCA}	η_t^{GA}	Task	η_t^{SVCA}	η_t^{GA}	Task	η_t^{SVCA}	η_t^{GA}
r_1	**1464.59**	1633.79	t_1	**106.46**	198.89	t_{16}	174.25	**21.65**	t_{31}	34.45	**7.10**
r_2	1247.24	**1028.90**	t_2	**19.52**	**19.52**	t_{17}	**39.82**	205.99	t_{32}	188.91	**24.70**
r_3	**358.76**	787.09	t_3	**147.98**	527.89	t_{18}	**10.38**	53.72	t_{33}	**52.98**	58.01
r_4	1297.75	**567.26**	t_4	**56.84**	61.98	t_{19}	**216.29**	347.11	t_{34}	**71.20**	500.29
r_5	**548.85**	3671.10	t_5	90.87	**37.74**	t_{20}	**245.70**	285.37	t_{35}	140.90	**41.24**
r_6	611.57	**562.36**	t_6	111.78	**90.18**	t_{21}	111.81	**86.73**	t_{36}	25.59	**13.82**
r_7	**270.81**	442.46	t_7	107.96	**30.21**	t_{22}	**27.25**	70.45	t_{37}	**154.48**	171.63
r_8	881.55	**209.03**	t_8	132.19	**21.75**	t_{23}	224.87	**46.63**	t_{38}	**95.25**	424.40
r_9	**69.98**	398.25	t_9	16.49	**8.05**	t_{24}	**15.40**	70.12	t_{39}	**31.93**	142.63
r_{10}	1602.31	**491.62**	t_{10}	165.12	**123.21**	t_{25}	78.18	**19.81**	t_{40}	33.72	**26.37**
r_{11}	464.78	**0.00**	t_{11}	278.10	**75.46**	t_{26}	27.29	**12.82**	t_{41}	**45.45**	165.61
r_{12}	1152.98	**133.58**	t_{12}	**29.49**	100.80	t_{27}	**59.54**	241.12	t_{42}	**47.36**	283.82
r_{13}	214.53	**0.00**	t_{13}	**7.28**	31.12	t_{28}	**38.79**	124.45	t_{43}	101.43	**62.57**
r_{14}	391.08	**0.00**	t_{14}	**20.54**	35.13	t_{29}	**34.23**	310.30	t_{44}	78.61	**36.28**
r_{15}	1079.34	**0.00**	t_{15}	176.41	**11.14**	t_{30}	98.28	**41.33**	t_{45}	56.85	**23.90**

of thousands of seconds. This time is strongly reduced by dividing the main problem into smaller clusters, which are allocated independently. More specifically, the computation time is in the range of hundreds of seconds in the case of the GA, and ranges between tens and hundreds of seconds for the SVCA. However, when the clustering is performed the allocation cost increases by 20% in comparison with that of centralized solutions. In any case, the significant reduction of computation time, around 98%, balances this loss of performance for problems requiring a fast allocation in few seconds.

5.5. Scalability analisys

Focusing on the SVCA and the GA, an additional comparison is performed to assess results with problem size. To this end, sets of problems with N ranging from 6 to 80 have been solved for $q_{\text{ini}} = 423$. The overall computation time $t_{\text{clu}} + t_{\text{all}}$ and clustering performance $V(\mathcal{C})$ of both approaches as a function of the problem size are represented in Fig. 7, where the results of the SVCA considering an exact evaluation of the Shapley value by (9) have also been included. Note that the use of the exact Shapley value becomes computationally intractable for schemes of around 20 players. This issue is solved by estimating the Shapley value using (13), which also outperforms the computation time of the GA for problems up to $N = 65$. In addition, performance results are very similar up to $N = 30$, from which the SVCA provides better outcomes.

Finally, a comparison of performance vs. time is presented in Fig. 8, for $N = 60$ and increasing values of q for the SVCA and the population size for the GA in order to exploit the available time represented in the $x-$axis of this figure. By increasing both parameters, the clustering performance improves but indeed the computation time gets worse. More specifically, our results show that the performance of the GA is very limited for small computation time requirements and improves as more time to solve the problem is allowed. Conversely, for the SVCA algorithm, Fig. 8 shows satisfactory performance results even with strong timing restrictions. The main reason for this performance is the use of qualitative rather than quantitative information by the SVCA, i.e., the use of a *relative* ranking of robots and tasks based on their Shapley values. Finally, as can be seen in Fig. 7, the computation time required for the proposed algorithm grows polynomially, being around 1 min for problems with 200 agents. Therefore, the size of problems that can be solved with the proposed method depends on the accessible computational resources, which will be employed to find the best possible solution within the available sampling time.

6. Conclusions

In this work, a game-theory based algorithm is presented for clustering multi-robot task allocation (MRTA) problems considering not only the distance but also the features of robots and tasks, which are measured by the Shapley value. The proposed

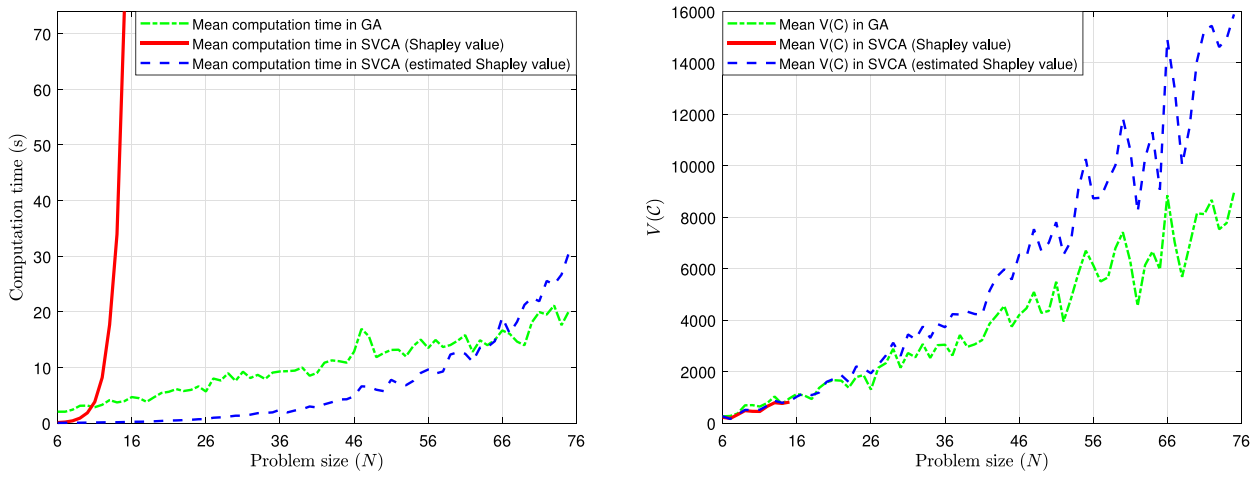

Fig. 7. Computation time (left) and performance (right) comparison between the SVCA and the GA.

algorithm groups the robots and tasks to balance the *aggregate* Shapley values in the resulting clusters. Also, this clustering algorithm can be applied in large problems using randomized methods such as the one proposed in [56,57] with satisfactory results in terms of computational burden and performance, outperforming other metaheuristic methods such as genetic algorithms (GA) and achieving a feasible solution much faster than centralized schemes. It is important to remark that the proposed method does not only provide a clustering, but also qualitative information regarding how useful a robot is or how demanding a task is.

In the current implementation, the proposed method can manage problems of about hundreds of agents. Therefore, there might be limitations in the number of players derived from the available time for the computations. Nevertheless, parallel computing techniques can be implemented to reduce not only the allocation duration but also the Shapley value calculation time because the algorithm in [56,57] allows to compute each sample independently. Furthermore, the new techniques being developed to compute the Shapley value of problems with thousands of players in machine learning applications might enhance the applicability of our method.

Future work will deal with *adaptative* approaches that allow recalculating the clustering cyclically for cases where robot and task features change or where there are different events such as unreachable objectives, dynamical appearance of new tasks, etc. Finally, *link-games* where players are the connections between robots and tasks will also be explored.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

All necessary data to reproduce the presented results are included along the manuscript.

Acknowledgments

This work has received funding from the H2020 ADG-ERC OCONTSOLAR Project under Grant 789051, and the Spanish MCIN/AEI/10.13039/501100011033 C3PO-R2D2 Project under Grant PID2020-119476RB-I00.

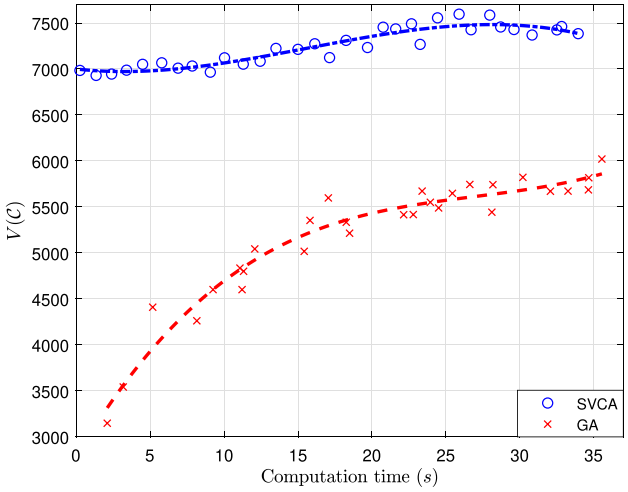

Fig. 8. $V(\mathcal{C})$ vs. computation time. Both algorithms, GA and SVCA, tend to improve with the computation time increase. In the case of GA, this fact allows considering a higher population size. For SVCA, larger q will produce more accurate Shapley value estimations.

References

[1] A. Brusell, G. Andrikopoulos, G. Nikolakopoulos, A survey on pneumatic wall-climbing robots for inspection, in: Proceedings of the 24th Mediterranean Conference on Control and Automation, MED, Athens, Greece, 2016, pp. 220–225.

[2] N.R. Zema, E. Natalizio, E. Yanmaz, An unmanned aerial vehicle network for sport event filming with communication constraints, in: Proceedings of the First International Balkan Conference on Communications and Networking, BALKANCOM, Tirana, Albania, 2017, pp. 1–5.

[3] P.S. Gohari, H. Mohammadi, S. Taghvaei, Using chaotic maps for 3D boundary surveillance by quadrotor robot, Appl. Soft Comput. 76 (2019) 68–77.

[4] J.J. Roldán, J. del Cerro, D. Garzón-Ramos, P. Garcia-Aunon, M. Garzón, J. de León, A. Barrientos, Robots in agriculture: state of art and practical experiences, in: A.J.R. Neves (Ed.), Service Robots, InTech, Rijeka, Croatia, 2017, pp. 67–90.

[5] Y. Wang, Z.-Y. Ru, K. Wang, P.-Q. Huang, Joint deployment and task scheduling optimization for large-scale mobile users in multi-UAV-enabled mobile edge computing, IEEE Trans. Cybern. 50 (9) (2020) 3984–3997.

[6] A. Farinelli, N. Boscolo, E. Zanotto, E. Pagello, Advanced approaches for multi-robot coordination in logistic scenarios, Robot. Auton. Syst. 90 (2017) 34–44.

[7] H. Huang, A.V. Savkin, M. Ding, C. Huang, Mobile robots in wireless sensor networks: A survey on tasks, Comput. Netw. 148 (2019) 1–19.

[8] A. Khamis, A. Hussein, A. Elmogy, Multi-robot task allocation: A review of the state-of-the-art, in: A. Koubâa, J.R. Martínez-de Dios (Eds.), Cooperative Robots and Sensor Networks 2015, in: Studies in Computational Intelligence 604, Springer, Cham, Switzerland, 2015, pp. 31–51.
[9] N. Seenu, R.M. Kuppan Chetty, M.M. Ramya, M.N. Janardhanan, Review on state-of-the-art dynamic task allocation strategies for multiple-robot systems, Ind. Robot 47 (6) (2020) 929–942.
[10] B.P. Gerkey, M.J. Matarić, A formal analysis and taxonomy of task allocation in multi-robot systems, Int. J. Robot. Res. 23 (9) (2004) 939–954.
[11] G.A. Korsah, A. Stentz, M.B. Dias, A comprehensive taxonomy for multi-robot task allocation, Int. J. Robot. Res. 32 (12) (2013) 1495–1512.
[12] H.W. Kuhn, The hungarian method for the assignment problem, Nav. Res. Logist. Q. 2 (1–2) (1955) 83–97.
[13] C. Nam, D.A. Shell, Assignment algorithms for modeling resource contention and interference in multi-robot task-allocation, in: Proceedings of the 31st IEEE International Conference on Robotics and Automation, ICRA, Hong Kong, China, 2014, pp. 2158–2163.
[14] E.G. Jones, M.B. Dias, A. Stentz, Time-extended multi-robot coordination for domains with intra-path constraints, Auton. Robots 30 (1) (2011) 41–56.
[15] X. Li, Z. Liu, F. Tan, Multi-robot task allocation based on cloud ant colony algorithm, in: Proceedings of the 14th International Conference on Neural Information Processing, ICONIP, Guangzhou, China, 2017, pp. 3–10.
[16] X.-S. Yang, Metaheuristic optimization: Algorithm analysis and open problems, in: Proceedings of the 10th International Symposium on Experimental Algorithms, SEA, Creete, Greece, 2011, pp. 21–32.
[17] M.B. Dias, R. Zlot, N. Kalra, A. Stentz, Market-based multirobot coordination: A survey and analysis, Proc. IEEE 94 (7) (2006) 1257–1270.
[18] F. Tang, L.E. Parker, A complete methodology for generating multi-robot task solutions using ASyMTRe-D and market-based task allocation, in: Proceedings of the 24th IEEE International Conference on Robotics and Automation, ICRA, Rome, Italy, 2007, pp. 3351–3358.
[19] R. Zlot, A. Stentz, Market-based multirobot coordination for complex tasks, Int. J. Robot. Res. 25 (1) (2006) 73–101.
[20] R.M. Zlot, An auction-based approach to complex task allocation for multirobot teams (Ph.D. thesis), The Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA, 2006.
[21] J. Turner, Q. Meng, G. Schaefer, A. Whitbrook, A. Soltoggio, Distributed task rescheduling with time constraints for the optimization of total task allocations in a multirobot system, IEEE Trans. Cybern. 48 (9) (2018) 2583–2597.
[22] W. Zhao, Q. Meng, P.W.H. Chung, A heuristic distributed task allocation method for multivehicle multitask problems and its application to search and rescue scenario, IEEE Trans. Cybern. 46 (4) (2016) 902–915.
[23] T. Rahwan, T.P. Michalak, M. Wooldridge, N.R. Jennings, Coalition structure generation: A survey, Artificial Intelligence 229 (2015) 139–174.
[24] F. Dörfler, M.R. Jovanović, M. Chertkov, F. Bullo, Sparsity-promoting optimal wide-area control of power networks, IEEE Trans. Power Syst. 29 (5) (2014) 2281–2291.
[25] Z. Wang, Z. Yu, C.L. Philip Chen, J. You, T. Gu, H.-S. Wong, J. Zhang, Clustering by local gravitation, IEEE Trans. Cybern. 48 (5) (2018) 1383–1396.
[26] J.M. Mulvey, M.P. Beck, Solving capacitated clustering problems, European J. Oper. Res. 18 (3) (1984) 339–348.
[27] M. Negreiros, A. Palhano, The capacitated centred clustering problem, Comput. Oper. Res. 33 (6) (2006) 1639–1663.
[28] J. Brimberg, N. Mladenović, R. Todosijević, D. Urošević, Solving the capacitated clustering problem with variable neighborhood search, Ann. Oper. Res. 272 (1–2) (2019) 289–321.
[29] O. Shehory, S. Kraus, Methods for task allocation via agent coalition formation, Artificial Intelligence 101 (1–2) (1998) 165–200.
[30] A. Dutta, V. Ufimtsev, A. Asaithambi, E. Czarnecki, Coalition formation for multi-robot task allocation via correlation clustering, Cybern. Syst. 50 (8) (2019) 711–728.
[31] M.U. Arif, Robot coalition formation against time-extended multi-robot tasks, Int. J. Intell. Unmanned Syst. 10 (4) (2022) 468–481.
[32] B. Peleg, P. Sudhölter, Introduction to the Theory of Cooperative Games, second ed., in: Theory and Decision Library C: Game Theory, Mathematical Programming and Operations Research, Springer, Heidelberg, Germany, 2007.
[33] R.P. Gilles, The Cooperative Game Theory of Networks and Hierarchies, in: Theory and Decision Library C: Game Theory, Mathematical Programming and Operations Research, Vol. 44, Springer, Heidelberg, Germany, 2010.
[34] W. Saad, Z. Han, M. Debbah, A. Hjørungnes, T. Başar, Coalitional game theory for communication networks, IEEE Signal Process. Mag. 26 (5) (2009) 77–97.
[35] G. Zhang, K. Yang, P. Liu, E. Ding, Achieving user cooperation diversity in TDMA-based wireless networks using cooperative game theory, IEEE Commun. Lett. 15 (2) (2011) 154–156.
[36] Z. Han, D. Niyato, W. Saad, T. Başar, A. Hjørungnes, Game Theory in Wireless and Communication Networks: Theory, Models, and Applications, Cambridge University Press, New York, USA, 2012.
[37] G.C. Chasparis, J.S. Shamma, Network formation: Neighborhood structures, establishment costs, and distributed learning, IEEE Trans. Cybern. 43 (6) (2013) 1950–1962.
[38] W. Saad, Z. Han, H.V. Poor, T. Başar, Game-theoretic methods for the smart grid: An overview of microgrid systems, demand-side management, and smart grid communications, IEEE Signal Process. Mag. 29 (5) (2012) 86–105.
[39] M. Marzband, R.R. Ardeshiri, M. Moafi, H. Uppal, Distributed generation for economic benefit maximization through coalition formation-based game theory concept, Int. Trans. Electr. Energy Syst. 27 (6) (2017) e2313.
[40] H. Jaleel, J.S. Shamma, Distributed optimization for robot networks: From real-time convex optimization to game-theoretic self-organization, Proc. IEEE 108 (11) (2020) 1953–1967.
[41] A.C. Chapman, R.A. Micillo, R. Kota, N.R. Jennings, Decentralised dynamic task allocation: A practical game-theoretic approach, in: Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems, AAMAS, Budapest, Hungary, 2009, pp. 915–922.
[42] G. Arslan, J.R. Marden, J.S. Shamma, Autonomous vehicle-target assignment: a game-theoretical formulation, J. Dyn. Syst. Meas. Control 129 (5) (2007) 584–596.
[43] J.M. Aguilar-López, R.A. García, A.J. Sánchez, A.J. Gallego, E.F. Camacho, Mobile sensor for clouds shadow detection and direct normal irradiance estimation, Sol. Energy 237 (2022) 470–482.
[44] J.G. Martin, J.M. Maestre, E.F. Camacho, Spatial irradiance estimation in a thermosolar power plant by a mobile robot sensor network, Sol. Energy 220 (2021) 735–744.
[45] L.S. Shapley, A value for n-person games, in: H.W. Kuhn, A.W. Tucker (Eds.), Contributions to the Theory of Games II. Annals of Mathematics Studies, Vol. 28, Princeton University Press, Princeton, New Jersey, USA, 1953, pp. 307–317.
[46] F.J. Muros, J.M. Maestre, C. Ocampo-Martinez, E. Algaba, E.F. Camacho, A game theoretical randomized method for large-scale systems partitioning, IEEE Access 6 (2018) 42245–42263.
[47] R. Lucchetti, S. Moretti, F. Patrone, P. Radrizzani, The Shapley and Banzhaf values in microarray games, Comput. Oper. Res. 37 (8) (2010) 1406–1412.
[48] N. Tarashev, K. Tsatsaronis, C. Borio, Risk attribution using the Shapley value: Methodology and policy applications, Rev. Finance 20 (3) (2016) 1189–1213.
[49] F.J. Muros, D. Saracho, J.M. Maestre, Improving supply quality in distribution power networks: A game-theoretic planning approach, Electr. Power Syst. Res. 213 (2022) 108666.
[50] S.S. Liao, J.-D. Zhang, R. Lau, T. Wu, Coalition formation based on marginal contributions and the Markov process, Decis. Support Syst. 57 (2014) 355–363.
[51] R. Narayanam, Y. Narahari, A Shapley value-based approach to discover influential nodes in social networks, IEEE Trans. Autom. Sci. Eng. 8 (1) (2011) 130–147.
[52] Y.-J. Zhang, A.-D. Wang, Y.-B. Da, Regional allocation of carbon emission quotas in China: evidence from the Shapley value method, Energy Policy 74 (2014) 454–464.
[53] V. Ginsburgh, I. Zang, Shapley ranking of wines, J. Wine Econ. 7 (2) (2012) 169–180.
[54] J.M. Maestre, D. Muñoz de la Peña, A. Jiménez Losada, E. Algaba, E.F. Camacho, A coalitional control scheme with applications to cooperative game theory, Optim. Control Appl. Methods 35 (5) (2014) 592–608.
[55] F.J. Muros, J.M. Maestre, E. Algaba, T. Alamo, E.F. Camacho, Networked control design for coalitional schemes using game-theoretic methods, Automatica 78 (2017) 320–332.
[56] J. Castro, D. Gómez, J. Tejada, Polynomial calculation of the Shapley value based on sampling, Comput. Oper. Res. 36 (5) (2009) 1726–1730.
[57] J. Castro, D. Gómez, E. Molina, J. Tejada, Improving polynomial estimation of the Shapley value by stratified random sampling with optimum allocation, Comput. Oper. Res. 82 (2017) 180–188.
[58] M.K. Tarkowski, P.L. Szczepański, T.P. Michalak, P. Harrenstein, M. Wooldridge, Efficient computation of semivalues for game-theoretic network centrality, J. Artificial Intelligence Res. 63 (2018) 145–189.
[59] J.M. Maestre, H. Ishii, A PageRank based coalitional control scheme, Int. J. Control Autom. Syst. 15 (5) (2017) 1983–1990.
[60] Y. Kwon, M.A. Rivas, J. Zou, Efficient computation and analysis of distributional Shapley values, in: Proceedings of the 24th International Conference on Artificial Intelligence and Statistics, AISTATS, Vol. 130, 2021, pp. 793–801, *virtual event*.
[61] R. Jia, D. Dao, B. Wang, F.A. Hubis, N.M. Gurel, B. Li, C. Zhang, C.J. Spanos, D. Song, Efficient task-specific data valuation for nearest neighbor algorithms, Proc. VLDB Endow. 12 (11) (2019) 1610–1623.

[62] A. Ghorbani, M. Kim, J. Zou, A distributional framework for data valuation, in: Proceedings of the 37th International Conference on Machine Learning, ICML, Vol. 119, 2020, pp. 3535–3544, *virtual event*.
[63] D. Fryer, I. Strümke, H. Nguyen, Shapley values for feature selection: The good, the bad, and the axioms, IEEE Access 9 (2021) 144352–144360.
[64] R. Pradhan, A. Lahiri, S. Galhotra, B. Salimi, Explainable AI: Foundations, applications, opportunities for data management research, in: Proceedings of the IEEE 38th International Conference on Data Engineering, ICDE, Kuala Lumpur, Malaysia, 2022, pp. 3209–3212.
[65] A. Ghorbani, J. Zou, Data Shapley: Equitable valuation of data for machine learning, in: Proceedings of the 36th International Conference on Machine Learning, ICML, Vol. 97, Long Beach, California, USA, 2019 pp. 2242–2251.
[66] Q. Zhu, Hidden Markov model for dynamic obstacle avoidance of mobile robot navigation, IEEE Trans. Robot. Autom. 7 (3) (1991) 390–397.
[67] M. Čáp, P. Novák, A. Kleiner, M. Selecký, Prioritized planning algorithms for trajectory coordination of multiple mobile robots, IEEE Trans. Autom. Sci. Eng. 12 (3) (2015) 835–849.
[68] H.W. Becker, J. Riordan, The arithmetic of bell and stirling numbers, Amer. J. Math. 70 (2) (1948) 385–394.
[69] K. Padmanabhan Panchu, M. Rajmohan, R. Sundar, R. Baskaran, Multi-objective optimisation of multi-robot task allocation with precedence constraints, Defence Sci. J. 68 (2) (2018) 175–182.
[70] A.T. Tolmidis, L. Petrou, Multi-objective optimization for dynamic task allocation in a multi-robot system, Eng. Appl. Artif. Intell. 26 (5–6) (2013) 1458–1468.
[71] J.G. Martin, J.R.D. Frejo, R.A. García, E.F. Camacho, Multi-robot task allocation problem with multiple nonlinear criteria using branch and bound and genetic algorithms, Intell. Serv. Robot. 14 (5) (2021) 707–727.
[72] J.G. Martin, R.A. García, E.F. Camacho, Event-MILP-based task allocation for heterogeneous robotic sensor network for thermosolar plants, J. Intell. Robot. Syst. 102 (2021) 1.
[73] Y. Zhang, L.E. Parker, Considering inter-task resource constraints in task allocation, Auton. Agents Multi-Agent Syst. 26 (3) (2013) 389–419.
[74] B.L. Brumitt, A. Stentz, GRAMMPS: A generalized mission planner for multiple mobile robots in unstructured environments, in: Proceedings of the 15th IEEE International Conference on Robotics and Automation, ICRA, Vol. 2, Leuven, Belgium, 1998, pp. 1564–1571.
[75] K. Al-Yafi, H. Lee, A. Mansouri, MTAP-MaSim: A multi-agent simulator for the mobile task allocation problem, in: Proceedings of the 18th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises, WETICE, Groningen, The Netherlands, 2009, pp. 25–27.
[76] B. Coltin, M. Veloso, Mobile robot task allocation in hybrid wireless sensor networks, in: Proceedings of the 23rd IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, Taipei, Taiwan, 2010 pp. 2932–2937.
[77] J.G. Martin, M. Hanif, T. Hatanaka, J.M. Maestre, E.F. Camacho, Predictive receding-horizon multi-robot task allocation with moving tasks, in: Proceedings of the 20th European Control Conference, ECC, London, UK, 2022, pp. 2030–2035.
[78] S. Giordani, M. Lujak, F. Martinelli, A distributed algorithm for the multi-robot task allocation problem, in: Proceedings of the 23rd International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE), Cordova, Spain, 2010, pp. 721–730.
[79] H.-L. Choi, L. Brunet, J.P. How, Consensus-based decentralized auctions for robust task allocation, IEEE Trans. Robot. 25 (4) (2009) 912–926.
[80] P.-a. Gao, Z.-x. Cai, L.-l. Yu, Evolutionary computation approach to decentralized multi-robot task allocation, in: Proceedings of the Fifth International Conference on Natural Computation, ICNC, Tianjian, China, 2009, pp. 415–419.
[81] R.J. Weber, Probabilistic values for games, in: A.E. Roth (Ed.), The Shapley Value: Essays in Honor of Lloyd S. Shapley, Cambridge University Press, Cambridge, UK, 1988, pp. 101–119.
[82] T. Hiller, Excluded coalitions and the distribution of power in parliaments, Appl. Econ. 48 (4) (2016) 321–330.
[83] T. Hiller, The effects of excluding coalitions, Games 9 (1) (2018) 1.
[84] P. Kudova, Clustering genetic algorithm, in: Proceedings of the 18th International Workshop on Database and Expert Systems Applications, DEXA, Regensburg, Germany, 2007, pp. 138–142.
[85] U. Maulik, S. Bandyopadhyay, Genetic algorithm-based clustering technique, Pattern Recognit. 33 (9) (2000) 1455–1465.
[86] M. Mitchell, An Introduction to Genetic Algorithms, MIT Press, Massachusetts, USA, 1998.

Javier G. Martin was born in Cadiz in 1990. Since September 2018, he combines his Ph.D. in the field of automation, robotics and telematics at the Department of Systems and Automation Engineering of the University of Seville, with working as a researcher for the ADG-ERC OCONTSOLAR Project. His research interests focus on robotics and predictive control.

Francisco Javier Muros received the Ph.D. on automation, robotics and telematics, summa cum laude and international mention, from the University of Seville in 2017. Since 2005, he works in the medium voltage south control center in Endesa, acquiring a wide experience in the power network real-time operation and management. He received a master's degree in project, construction and maintenance of high voltage electrical transmission from the Comillas Pontifical University, Madrid in 2014. Currently, he combines his work at Endesa with working as a part-time lecturer at the Loyola University Andalusia; also, he is an active researcher of the Department of Systems and Automation Engineering at the University of Seville. He has participated in the European Union OCONTSOLAR and DYMASOS Projects and in several national and regional projects. He has authored or co-authored more than 30 publications, highlighting 10-JCR journal papers and the book "Cooperative Game Theory Tools in Coalitional Control Networks" (Springer, 2019). His research interests focus on cooperative and noncooperative game theory and coalitional and distributed control.

José María Maestre got his PhD on automation and robotics from the University of Seville, where he works as full professor. He has also worked in other universities as TU Delft, University of Pavia, University of Keio, and Tokyo Institute of Technology. His main research interest is the control of distributed cyber-physical systems. He has (co-)authored around two hundred journal and conference papers and has (co-)edited the books "Service robotics within the Digital Home: Applications and Future Prospects" (Springer, 2011), "Distributed Model Predictive Control Made Easy" (Springer, 2014), and "Domótica para Ingenieros" (Paraninfo, 2015). Also, he has (co-)authored several books, including "A Programar se Aprende Jugando" (Paraninfo, 2017) and "Sistemas de Medida y Regulación" (Paraninfo, 2018).

Eduardo F. Camacho received the Ph.D. degree in electrical engineering from the University of Seville, where he is now a full professor with the Department of System and Automation Engineering. He is author of "Model Predictive Control in the Process Industry" (1995), "Control e Instrumentación de Procesos Químicos" (1997), "Advanced Control of Solar Plants" (1997), "Model Predictive Control" (1999, 2004), "Control of Dead-time Processes" (2007) and "Control of Solar Systems" (2012, translated into Chinese in 2014). He has served on various IFAC technical committees and chaired the IFAC publication Committee from 2002–2005. He was the president of the European Control Association (2005–2007) and chaired the IEEE/CSS International Affairs Committee (2003–2006), Chair of the IFAC Policy Committee, a member of the IEEE/CSS Board of Governors, and a member of the IFAC Council. He has acted as evaluator of projects at national and European level and was appointed Manager of the Advanced Production Technology Program of the Spanish National RD Program (1996–2000). He was one of the Spanish representatives on the Program Committee of the Growth Research program and expert for the Program Committee of the NMP research priority of the European Union. He has carried out reviews and editorial work for various technical journals and many conferences. He has been one of the Editors of the IFAC Journal, Control Engineering Practice, Editor-at-Large of the European Journal of Control and Subject Editor of Optimal Control: Methods and Applications. Dr. Camacho is an IEEE and IFAC Fellow. He was Publication Chair for the IFAC World Congress 2002, General Chair of the joint 44th IEEE CDC-ECC 2005, and co-General Chair of the joint 50th IEEE CDC-ECC 2011. He has been awarded an Advanced Grant by the European Research Council for the OCONTSOLAR Project, consisting of integrating solar radiation sensors mounted in drones to control solar plants.