Solving the Multi-Robot Task Allocation Problem using Firefly, Artificial Bee Colony and Quantum Genetic Algorithms

Farouq Zitouni[1], Saad Harous[2] and Ramdane Maamri[3]
[1] KASDI Merbah university – Ouargla, Algeria – farouq.zitouni@univ-constantine2.dz
[2] Al-Ain University – Al-Ain city, UAE – harous@uaeu.ac.ae
[3] Constantine 2 University – Constantine, Algeria – ramdane.maamri@univ-constantine2.dz

***Abstract*—The Multi-Robot Task Allocation problem of is the situation where we have a set of tasks and a number of robots, and each task have to be assigned to the appropriate robots that optimize some criteria, e.g. allocate the maximum number of tasks. We present an effective solution to address this problem. It implements a two-stage methodology: first, a global allocation using firefly algorithm, then a local allocation combining artificial bee colony optimization and quantum genetic algorithms. Experimental results show the effectiveness of the proposed solution in terms of the number of allocated tasks and the allocation time.**

***Index Terms*—Multi-Robot System, Task Allocation, Firefly Algorithm, Artificial Bee Colony Optimization, Quantum Genetic Algorithms.**

I. Introduction

Nowadays, Multi-Robot Systems (MRS) are almost in every aspect of our daily lives, and intensive research is being done to achieve their opportunities. The main motivation behind such attention is because they make it easy to solve many complex problems, such as industrial applications, surveillance, and rescue missions in environments hit by natural disasters [1].

Also, it must be mentioned that the design of a MRS must consider interactions between its components – i.e. robots, otherwise, it risks to produce a system with limited and non-deterministic performance [2]. Thereby, one of major coordination problems, that researchers must solve in this field, is the Task Allocation – named Multi-Robot Task Allocation (MRTA) [3].

To illustrate the correlation between task allocation, multi-robot system, and industrial applications, we give the following scenario: Think of An online sale company that sells an article every hour; a robot at the warehouse receives the order, finds the item, packages, prepares and sends it through an appropriate postal service. What happens if the company sells 20 items every hour? Every minute? Each second? In 2013, Amazon's online sale site sold 36.8 million items on a particularly popular shopping day. With 426 items ordered per second that day, a single robot would struggle to meet and follow all orders. If the warehouse used a team of robots i.e. a multi-robot system, each robot should plan an efficient path, e.g. the shortest one, through the warehouse to retrieve the items to be shipped without colliding with others and without taking their objects.

A. Definition of the MRTA problem

The MRTA problem can be defined as follows: We assume that we have some tasks to be performed and robots that are capable of achieving these tasks: for example, we assume a task that requires capturing temperature and humidity of a given place; if a robot can capture at least one of these two measurements, then we say it can achieve this task. Considering tasks to be allocated, the system should find the robots are adequate to execute them. These assignments must optimize the cost function taken into account, e.g. minimize completion time or makespan. It is worth pointing out that the cost function is global, i.e. it considers the achievement of all task. So, the goal is to coordinate robots' behaviors and find an optimal way to allocate all tasks [4], i.e. usually, the task accomplishment requires sensors and actuators, if a sensor or actuator is offered by several robots, the coordination of robots' behaviors means how these resources are used: e.g. the order and period of use, etc. As illustrated in Figure 1 the MRTA problem can be formulated as an optimal assignment problem, where the objective is to optimally allocate a set of tasks to a set of robots while optimizing a given cost function, e.g. minimize the consumed energy, and satisfy some constraints.

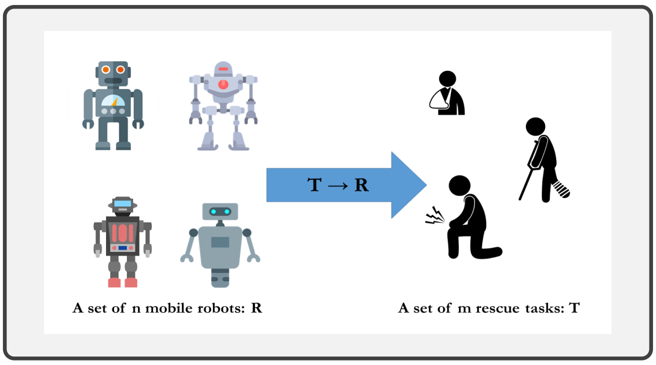

Figure 1: Multi-Robot Task Allocation problem.

In this problem, we consider a set of mobile robots: $R = \{r_1, r_2, \ldots, rn\}$ and set of feasible tasks: $T = \{t_1, t_2, \ldots, t_m\}$. We also consider a set of robots utilities: u_{ij} is the robot utility i to execute task j. The basic mathematical formulation of this problem is described in Table I [5].

Table I: The basic mathematical formulation of MRTA problem

optimize	$\sum_{i \in N} \sum_{j \in M} x_{ij} u_{ij}$		
subject to			
1)	$\sum_{j \in M} x_{ij} = 1$	,	$\forall i \in N$
2)	$\sum_{i \in N} x_{ij} = 1$	,	$\forall j \in M$
3)	$x_{ij} \in \{0, 1\}$	,	$\forall i \in N, \forall j \in M$

Where

x_{ij}	:	$x_{ij} \in \{0, 1\}$ is an indicator. If $x_{ij} = 1$, then task j is allocated to robot i, otherwise it is not.
N	:	robots' set.
M	:	tasks' set.
u_{ij}	:	cost function to be optimized.

Also, other constraints may be considered, e.g. spatial, energetic, and temporal. In linear programming, this formulation is called assignment problem – can be easily solved using an adequate algorithm, e.g. Hungarian method [6]: a combinatorial optimization algorithm that solves the assignment problem in polynomial time. Generally, exact methods do not capture constraints imposed by real-life problems, hence the need to rethink them for the treatment of complex cases.

A task is a sub-goal of the system it belongs to, and then the accomplishment of all tasks is considered the ultimate goal of the system. It is executed in a dependent manner – e.g. after another task, or in an independent manner – i.e. no constraint is imposed. It is discrete – e.g. move an object from point A to point B, continuous – e.g. monitor the entry of a bank, or hybrid – e.g. move an object from point A to point B while monitoring its environment to avoid collisions. To allocate tasks, every robot must calculate a numerical value for each one. Intuitively, this measurement is called utility – i.e. it expresses cost, gain, distance, time, or their combination.

B. *Definition of bids*

Solving the MRTA problem, in an optimal way, is to find assignments between the given set of tasks to a subset of the available robots, while optimizing a considered numerical measurement. Commonly, this value is named bid [7]. To approach this concept, we give some tasks that must be allocated to some robots. We assume that every robot is capable to compute a numerical value for each task – noted u_{ij}. Thus, the bid of a robot r for a task t is computed by Equation (1).

$$u_{rt} = \begin{cases} q_{rt} - c_{rt} & \text{, if } r \text{ is capable of executing } t \\ \infty & \text{, otherwise} \end{cases} \tag{1}$$

Where

q_{rt}	:	quality of performing t by r.
c_{rt}	:	cost of performing t by r.

Also, it is important to note that the bid of a robot for a task can depend on other robots' bids of the same environment [8]. Thereby, the previous equation can be generalized to manipulate robots' and tasks' subsets. We assume that R and T are two robots' and tasks' subsets – $|R| \geq 1$ and $|T| \geq 1$. Thus, Equation (1) can be reformulated as follows.

$$U_{rt} = \begin{cases} Q_{RT} - C_{RT} & \text{, if robots in } R \text{ are capable of executing tasks in } T \\ \infty & \text{, otherwise} \end{cases} \tag{2}$$

Where

Q_{RT}	:	quality of performing tasks in T by robots in R.
C_{RT}	:	cost of performing tasks in T by robots in R.

In practice, we find several variants of MRTA problems, and the choice of algorithms to solve them depend on the faced configuration. In the next section, we give three taxonomies used for the classification of MRTA problems.

C. *Taxonomies for MRTA problems*

In the literature, we find three taxonomies to categorize MRTA problems. Their purpose is to propose a classification of task allocation situations into classes, so that real-life problems can be projected onto one of them – often a mathematical formulation is given. For simplicity, we shall name these taxonomies: taxonomy A, B, and C, respectively.

*1) **Taxonomy A**:* The authors of [7] categorize MRTA problems by proposing a classification which is articulated around three axes. The first one distinguishes robots according to their abilities to be assigned either to a single task or several tasks at a time. The second one distinguishes tasks according to their need to be allocated either to a single robot or to several robots at the same time. The third one takes into account how tasks are assigned to robots considering either only current or both current and future information.

The authors [7] defined two points at each axis as follows. Axis 1: Single-Task Robots (ST), Multi-Task Robots (MT), axis 2: Single-Robot Tasks (SR), Multi-Robot Tasks (MR), and axis 3: Instantaneous Assignment (IA), Time-extended Assignment (TA). That is to say, Single-Task Robots (ST) vs. Multi-Task Robots (MT) i.e. the number of tasks that a robot is capable of executing simultaneously; Single-Robot Tasks (SR), Multi-Robot Tasks (MR) i.e. the number of robots required to execute a task simultaneously; Instantaneous Assignment (IA) vs. Time-extended Assignment (TA) i.e. allocations are done without considering or considering future tasks.Thus, eight problem classes are distinguished, i.e. ST-SR-IA, ST-SR-TA, ST-MR-IA, ST-MR-TA, MT-SR-IA, MT-SR-TA, MT-MR-IA, and MT-MR-TA. Despite its good coverage of most MRTA problems, however, this taxonomy does not capture problems with interrelated utilities, and temporal constraints on tasks and robots [7]. Hence the need to propose a taxonomy that addresses these limitations.

*2) **Taxonomy B**:* The authors of the work [8] modified taxonomy A by adding a higher level which expresses interdependence degrees between tasks and robots. This taxonomy is a hierarchy with two levels. The second one includes the classes proposed in Taxonomy A. The first one encompasses four interdependence degrees between tasks and robots as follows. One, "No Dependencies (ND)": The utility of a robot for a task does not depend on anything – it covers ST-SR-IA and ST-SR-TA problem classes. Two, "In-Schedule Dependencies (ID)": The utility of a robot for a task depends only on its schedule – it covers ST-SR-TA, MT-SR-IA, and MT-SR-TA problem classes. Three, "Cross-Schedule Dependencies (XD)": The utility of a robot for a task does not only depend on its schedule but also on other robots' schedules, and schedules are static – it covers the eight problem classes. Four, "Complex Dependencies (CD)": The utility of a robot for a task does not only depend on its schedule and on other robots' schedules, but in this case schedules are dynamic – it covers the eight problem classes.

*3) **Taxonomy C**:* The authors of [9] have modified taxonomy A by splitting the "Time-extended Assignment (TA)" point into two sub-points. This action allows the introduction of temporal constraints on tasks as follows. First, the sub-point "TA:TW" which allows to consider temporal constraints in the form of Time Windows. Second, the sub-point "TA:SP" which makes it possible to consider Synchronization and Precedence constraints.

D. Contribution and Paper organization

The main contribution of this paper is the proposal of a two-stage solution to solve the MRTA problem. In the first stage the firefly algorithm is applied for the global allocation, and in the second stage a hybridization – between artificial bee colony and quantum genetic algorithms – is applied for the local allocation. On one hand, the MRTA problem is an optimization problem where a given objective function has to be minimized (or maximized) satisfying some constraints. On the other hand, we also know that exact methods – used to solve such problems – quickly become ineffective when the search space increases [10], hence the need for heuristics. The firefly algorithm has a major advantage over other heuristics because it can subdivide a population of robots into subgroups using its abilities of attraction and attractiveness. Then each subgroup is assigned to a task for the second stage (local allocation). This property allows us to handle several tasks at once, which increases the efficiency of our solution. It is worth pointing out that the choice of a heuristic for solving an optimization problem must combine two main components: exploration and exploitation of the research space. If they are well adjusted, these two components allow us – in the second stage of our solution – to allocate each task to best robots, i.e. best robots belonging to a subgroup. Optimization solution using artificial bee colony is known for its well exploration and quantum genetic algorithm is well known for its well exploitation of the search space. Then their combination results in a more efficient heuristic.

The remainder of the paper is organized as follows. In section II, we give an overview of previous work addressing the MRTA problem. In Section III, we present the proposed solution to treat this problem and explain its steps. In section IV, we simulate our solution, compare it to another approach [11], and discuss the obtained results. In section V, we give a conclusion and some perspectives.

II. Related Work

We discuss previously proposed approaches to address the MRTA problem. These solutions are listed taking into consideration the first level of taxonomy B, i.e. they are divided into four groups. It should be noted that this list is not exhaustive, however, it covers those frequently cited in the literature [4].

In the first group, we find several approaches in the literature [7]. Particularly, they dealt with ND[ST-SR-IA] problems. We mention the work in [12] that uses potential fields, and works in [13], [14] that use auction-based approaches. In the second group, we find several works. Generally, they focused on ID[ST-SR-TA] problems. The work in [15] uses TSP and mTSP to treat this problem. The solution provided in [16] is a good solution for the routing problem using mixed linear programming. Due to their distributed nature, auction-based approaches are also used to address this problem [17], [18], [19], [20].

In the third group, there are several works. Among them M+ system [21] which addressed the instantaneous task allocation by using a market system with precedence constraints. The work [22] proposes a solution that manipulates constraints on tasks and a market-economy method. The work [23] addresses the routing problem by proposing three methodologies – robots' teams must perform scientific missions. A solution that uses quantum genetic algorithms and reinforcement learning is proposed in [24]. The case of simple task scheduling is considered in [25] which uses the heuristic: "task x must be accomplished n second(s) before the starting of task y". Another way to solve this problem is to bring it to a coalition formation problem [26], [27]. The approach proposed in [28] enhances the works in [26], [27] by minimizing communication rates and imposing constraints on agents' capabilities. The authors of [29], [30] use auctions and coalitions to address this problem. The Framework [31] proposes a solution that imposes the constraint of shared resources, i.e. communication mediums and processors. Finally, the solutions proposed in [32], [33] addressed the problem where robots must consider current and future information for task allocations.

In the fourth group, there are fewer works because of the increased difficulty of the problems and the lack of mathematical formulations [8]. The work [34] addresses this problem while considering both current and future information for task allocations – the approach has been applied to manage an environment hit by a natural disaster. The approach presented in [35] uses coalition formation – this solution was originally named ASyMTRe in its centralized version and latter ASyMTRe-D in its distributed version. Finally, we mention the work [36] that modified [17] and dealt with both current and future assignments.

III. Proposed Solution

An auction-based solution, named FA-QABC-MRTA (Firefly Algorithm-Quantum Artificial Bee Colony-Multi-Robot Task Allocation), is presented to solve the MRTA problem. Auction-based approaches are frequently used to address this problem because of their advantages, e.g. the simplicity of cost function satisfaction [37]. These approaches are based on auctions, which consist of an auctioneer, bidders, and goods. If a MRTA problem is taken into account, a particular robot is an auctioneer, rest of robots are the bidders, and tasks are the goods. We use the Contract Net Protocol [38] for communications.

A. *Roles of system actors and assumptions*

The auctioneer communicates tasks to be allocated to the bidders and receives their offers. After that, it decides the best assignments between bidders and tasks and finally notifies the concerned ones. Although the use of an auctioneer can be a bottleneck – e.g. if it breaks down then the system will break down too, but it greatly reduces the number of exchanged messages and keeps a global view of the current system state. Also, this solution has the advantage of calculations' sharing on bidders (scalability), which would make it possible to maintain the progression of assignments even if one of them breaks down (robustness) [37].

The bidders calculate utilities for tasks based on information they have. So, the overall cost function is divided into sub-functions – estimated in a decentralized and independent manner by bidders. In our solution, the execution of a task simply means the presence of resources it requires – i.e. a task is said to be executed if and only if resources it requires are all available. The resources are made available by bidders, and are usually sensors and actuators.

We assume the class of problems ST-MR-IA, i.e. each bidder is allocated at most to one task at a time, some tasks are allocated to several bidders at a time, and tasks are assigned to bidders considering only current information. We give two non-empty sets B and T – bidders and tasks, respectively. We suppose that all tasks are known beforehand or discovered gradually. Each bidder $b \in B$ is defined by a sextuplet $\langle R_b, C_b, v_b, e_b, a_b, p_b \rangle$. The parameter R_b is a binary vector representing its resources and C_b is a vector representing resource costs. The other parameters represent its speed, energy level, aging factor, and spatial position, respectively. Each task $t \in T$ is defined by a triplet $\langle R_t, d_t, p_t \rangle$. The parameter R_t is a binary vector representing resources it requires. The other parameters represent its duration, and spatial position, respectively. Next, we present and explain the different algorithms proposed to address the MRTA problem.

B. *Actors' behaviors*

The following algorithms show behaviors of the auctioneer and bidders. Algorithms 1 and 2 have two sub-behaviors – i.e. "Periodic Behavior" and "Cyclic Behavior". The first one includes the operations that are run periodically – i.e. every n seconds. The second one includes the operations that are executed after a message is received. These behaviors are explained in the following sections.

Algorithm 1: the auctioneer behavior.

```
While The Auctioneer Is Alive
    responses ← ∅;{It contains bidders' responses.}
    {This behavior is executed periodically.}
    Periodic Behavior
        {If the condition is met, it means that some tasks
         are available for allocation.}
        if (|T| ≠ 0) then
            foreach b ∈ B do
                {A message contains the receiver, subject,
                 and content, respectively.}
                sendMessageTo(b,"CHOOSE-A-
                TASK",T);
            end
        end
    end
    {This behavior is executed after a message is
     received.}
    Cyclic Behavior
        responses ← responses
         ∪{receiveMessageFrom(b)};
        {If the condition is met, it means that all bidders'
         responses are received.}
        if (responses.isAll() = true) then
            foreach t ∈ T do
                if (t.canBeAllocated() = true) then
                    {B' is a set of robots (B' ⊆ B).}
                    {An allocation is computed using
                     Algorithm 3.}
                    B' ← computeAnAllocationFor(t);
                    T ← T/{t};
                    foreach b ∈ B' do
                        sendMessageTo(b,"NOTIFY",t);
                    end
                end
            end
            responses ← ∅;
        end
    end
end
```

The task allocation process begins when the auctioneer announces to bidders the availability of tasks to be allocated – "Periodic Behavior" of Algorithm 1. To do this, it broadcasts a message to all bidders – which contains the resources required by tasks and their position.

If a message is received – "Cyclic Behavior" of Algorithm 2, each bidder chooses a task to execute, calculates its cost, and replies to the auctioneer – "Periodic Behavior" of Algorithm

2. To choose a task, a bidder executes the firefly algorithm [39], [40] on received tasks, and gradually converges to one of them. To converge to a task, we use Eqs. (3) and (4). Once a task has been selected, the bidder estimates its cost for each resource it offers using Equation (5). Finally, the bidder sends its result to the auctioneer. The symbol $\perp$ means that the bidder sends a special notification to the auctioneer informing it that it did not select any task.

Algorithm 2: the bidder behavior.

```
While The Bidder Is Alive
    T ← ∅;{It contains tasks available for allocation.}
    {This behavior is executed periodically.}
    Periodic Behavior
        if (|T| ≠ 0) then
            if (state = "AVAILABLE") then
                {n is the number of iterations.}
                for i ← 1 to n do
                    foreach t ∈ T do
                        Converge the bidder to the task t
                          using Equation (3) and (4).
                    end
                end
                t ← choose the closest task to the bidder;
                c ← estimate costs of offered resources
                  using Equation (5);
                {A message contains the subject, chosen
                  task, and costs, respectively.}
                  sendMessageToTheAuctioneer("CHOSEN-
                  TASK",t,c);
            else
                {An empty message is sent.}
                sendMessageToTheAuctioneer("CHOSEN-
                  TASK",⊥,⊥);
            end
            T ← ∅;
        end
    end
    {This behavior is executed after a message is
      received.}
    Cyclic Behavior
        message ← receiveMessageFromTheAuctioneer();
        if (message.isEmpty() ≠ true) then
            if (message.getSubject() =
              "CHOOSE-A-TASK") then
                T ← message.getContent();
        end
    end
end
```

$$p_b = p_b + e^{-\gamma \times (d_{bt})^2} \times (p_t - p_b) + \alpha(rand - 0.5) \quad (3)$$

$$d_{bt} = \delta \|p_t - p_b\| + (1 - \delta) H(R_t, R_b) \quad (4)$$

$$u^{bt}_{r \in \Delta_{bt}} = [\delta c^b_r + (1 - \delta)(\frac{v_b}{\|p_t - p_b\|})^{\frac{e_b}{a_b}}]|\Delta_{bt}| \quad (5)$$

Where

p_b	:	position of robot b.
p_t	:	position of task t.
$e^{-\gamma \times (d_{bt})^2} \times (p_t - p_b)$	:	attraction degree of b by t.
γ	:	attraction variation – its value is very important for the algorithm convergence speed and the behavior of b ($\gamma \in [0.01, 100]$).
α	:	environment noise – its value affects the visibility of tasks by bidders ($\alpha \in [0, 1[$).
$rand$	:	a random value ($rand \in [0, 1]$).
d_{bt}	:	weighted distance between b and t , it is given by Equation (4).
$\|p_t - p_b\|$	:	Euclidean distance between b and t.
$H(R_t, R_b)$	:	Hamming distance between the vectors R_t and R_b.
δ	:	parameter serves to balance the two distances ($\delta \in [0, 1]$).
Δ_{bt}	:	common resources between b and t – i.e. the resources offered by b to t.

When all bidders' responses are received, the auctioneer calculates an allocation for each selected task – "Cyclic Behavior" of Algorithm 1. To do this, it considers for each task all the resources offered by the bidders, then it combines the artificial bee colony and quantum genetic algorithms to calculate an allocation for it. At the end of this stage, bidders concerned by the found allocations are notified. Algorithm 3 presents the steps of this combination.

C. ABC and QGA for task allocation

To explain the instructions of Algorithm 3, we use an illustrative example and clarify algorithm equations. It hybridizes advantages of two methods, well known for their efficiency, frequently used to solve optimization problems, artificial bee colony [41] and quantum genetic [42] algorithms.

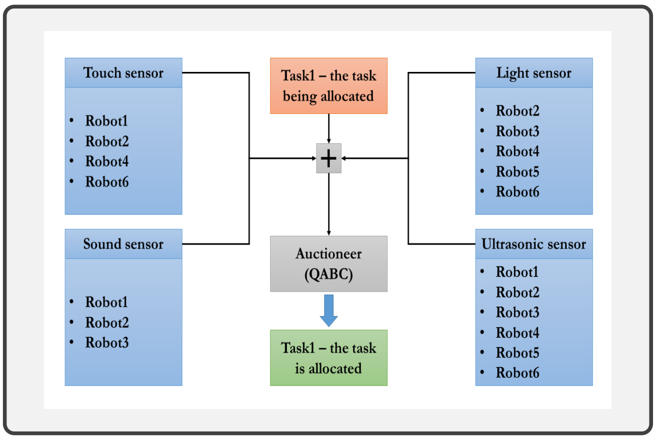

Figure 2: A scenario of a MRTA example.

Figure 2 depicts a simple scenario of a MRTA problem. In this example, we want to allocate Task1 – it requires four sensors, i.e. Touch, Sound, Light, and Ultrasonic. These sensors are offered by six bidders, i.e. Robot1, Robot2, Robot3, Robot4, Robot5, and Robot6. To each bidder sensor is assigned a real valuation representing its cost – calculated by Equation (5). Therefore, the objective is to find a subset, i.e. an allocation, satisfying the following three constraints.

1) The cost of this allocation must be minimal among all allocations,
2) The cardinality of this allocation is equal to the number of sensors required by the task taken into account – in the previous example, the cardinality is equal to 4, and
3) Resource repetitions in an allocation are not allowed, i.e. only one instance of each resource is required.

Generally, this kind of problems can be solved easily and optimally in linear programming – we just use an exact method. However, if the number of bidders and resources is large, then exact methods become very quickly ineffective – with the previous example, we have to consider $2^{4+3+5+6}$ different subsets and choose the allocation that simultaneously satisfies the previous constraints and its cost is optimal.

As a result, we propose to combine the powers of Artificial Bee Colony (ABC) and Quantum Genetic Algorithms (QGA) to address this optimization problem. The MRTA problem being addressed here can be modeled using the formulation illustrated in Table II. The term $u^{bt}_{r\in\Delta_{bt}}$ represents costs of resources offered by bidders. R_t is the set of resources needed by the task t. Constraint 1 gives the range of allowed values for costs. Constraints 2 and 3 ensure that two bidders never offer the same resource at the same time. Constraint 4 ensures that the cardinality of an allocation is equal to the number of resources required by the task. Constraint 5 is an indicator: if bidder b offers a resource r to a task t then its value is 1, otherwise its value is 0.

Table II: The mathematical formulation of our MRTA problem

min	$\sum_{r\in R_t} u^{bt}_{r\in\Delta_{bt}}$			
subject to				
1)	$\forall t \in T$	$\forall b \in B$		$u^{bt}_{r\in\Delta_{bt}} \in [l_r, u_r]$
2)	$\forall t \in T$	$\forall b, b' \in B$	$\forall r, r' \in R_t$	$(x^{bt}_r - x^{b't}_{r'} = 0) \rightarrow r \neq r'$
2)	$\forall t \in T$	$\forall b \in B$	$\forall r, r' \in R_t$	$(x^{bt}_r - x^{bt}_{r'} = 0) \rightarrow r \neq r'$
2)	$\forall t \in T$	$\forall b \in B$		$\sum_{r\in R_t} x^{bt}_r = \lvert R_t \rvert$
4)	$\forall t \in T$	$\forall b \in B$	$\forall r \in R_t$	$x^{bt}_r \in \{0, 1\}$

*1) **Initialization of individuals**:* A quantum population of food sources is initialized – i.e. individuals, candidate solutions, or allocations. Each individual contains several quantum chromosomes – i.e. the number of resources required by the task to be allocated. The number of genes in each chromosome is equal to the number of bidders that offered the corresponding resource – chromosomes' lengths are not necessarily identical. Figure 3 shows the structure of an individual, i.e. a candidate solution, of the previous example. We consider N the population size.

Algorithm 3: ABC and QGA for task allocation.

Randomly initialize a quantum population F^p_{ij} of n food sources using Equation (6);
$i \leftarrow 0$;
Make M^p_{ij} using Equation (7) – measure of every food source $F^p_{ij} \rightarrow M^p_{ij}$;
Evaluate every M^p_{ij} using constraints c1, c2, c3, and c4 – fitness function;
Memorize the best food source;
while *stopping criteria is not met* **do**
    Employed Bee Stage
        Perform an update process for each food source using Equation (9);
    end
    Onlooker Bee Stage
        Randomly select food sources using Equation (10);
        Perform an update process for each food source using Equation (9);
    end
    Rotation;
    Mutation;
    $i \leftarrow i + 1$;
    Make M^p_{ij} using Equation (7) – measure of every food source $F^p_{ij} \rightarrow M^p_{ij}$;
    Evaluate every M^p_{ij} using Equation (8) – fitness function;
    Memorize the best food source;
    Scout Bee Stage
        Select one of the most inactive food sources;
        Replace it by a new randomly generated one using Equation 6;
    end
end

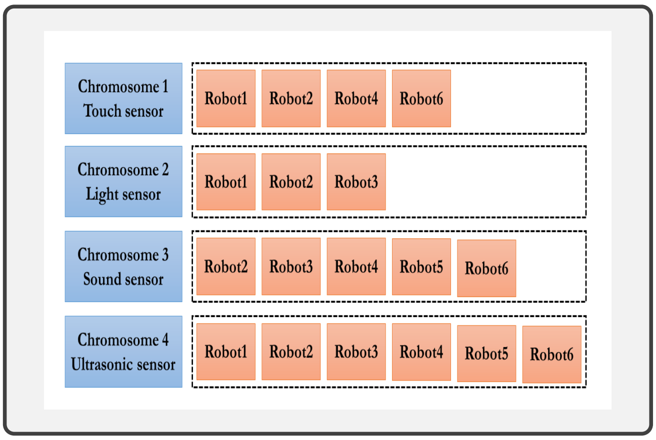

Figure 3: An example of an individual.

This individual contains four chromosomes – the task requires four resources, and the chromosomes' lengths are not equal. In general, a quantum chromosome is a $2 \times m$ matrix, where m is the number of genes in each chromosome. For example, the chromosome 3 in Figure 3 is a matrix 2×5 – we have five robots that offer the corresponding resource, i.e. the Light sensor. Equation (6) is used for individuals' initialization [42].

$$\begin{bmatrix} \alpha_1 & \dots & \alpha_m \\ \beta_1 & \dots & \beta_m \end{bmatrix} = \left[\begin{pmatrix} \cos(\theta_1) & -\sin(\theta_1) \\ \sin(\theta_1) & \cos(\theta_1) \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right.$$

$$\left. \dots \begin{pmatrix} \cos(\theta_m) & -\sin(\theta_m) \\ \sin(\theta_m) & \cos(\theta_m) \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \right] \quad (6)$$

*2) **Measurement and evaluation of individuals**:* To choose the best individual of a population – i.e. the best allocation, first for each individual we project its chromosomes on a binary space, then calculate its fitness value. To project chromosomes of an individual, we use Equation (7) [42] – this operation is called measurement or observation.

$$\begin{cases} y_i = 0 & , \quad m(\alpha) \leq |\alpha_j|^2 \\ y_i = 1 & , \quad m(\alpha) > |\alpha_j|^2 \end{cases} \quad (7)$$

Where

$m(\alpha)$: a random number in the interval $[0, 1]$.

So, each quantum gene $\begin{pmatrix} \alpha_j \\ \beta_j \end{pmatrix}$ becomes a binary gene $y_j \in \{0, 1\}$. Then, the fitness value of each allocation is calculated respecting the following constraints.

c1: If a binary chromosome q of an allocation p contains several 1 – that is: many robots offer the same resource, then its fitness value is $fit_{p,q} = +\infty$.

c2: If a binary chromosome q of an allocation p does not contain any 1 – that is: no robot offers the corresponding resource, then its fitness value is $fit_{p,q} = +\infty$.

c3: If a binary chromosome q of an allocation p contains exactly 1 – that is: one robot offers the need resource, then its fitness value is calculated using Equation (8) [41]. The parameter u_q^{bt} is the cost of resource q offered by bidder b to task t.

$$fit_{p,q} = \begin{cases} \frac{1}{u_q^{bt}} & , \quad u_q^{bt} \geq 0 \\ 1 + abs(u_q^{bt}) & , \quad u_q^{bt} < 0 \end{cases} \quad (8)$$

c4: Finally, the fitness value of an allocation is equal to the sum of its chromosomes' fitness values. The position of the best allocation is saved.

*3) **Hybridization of ABC and QGA operations**:* We explain the instructions of Algorithm 3 "while" loop. These instructions combine ABC and QGA operations to address the MRTA problem, and find the best allocation for a given set of tasks. During each iteration the following steps are executed.

1) In this step (from line 7 to 9), each employed bee calculates a new allocation V_i in the neighboring of its current allocation X_i and evaluates its quality. To do this, Equation (9) is used. The term X_k is a random allocation. The parameter j is a random index ($j \in [1, N]$). The parameter Φ_{ij} is a random number in the range $[-1, 1]$. If new allocation quality is better than the current one, the employed bee will move towards the new one – using a greedy selection. Otherwise, it stays in its current position.

$$V_{ij} = X_{ij} + \Phi_{ij}(X_{ij} - X_{kj}) \quad (9)$$

2) At the end of the previous step, employed bees share their allocation qualities with onlooker bees. Now (from line 10 to 13), each onlooker bee chooses an allocation using a roulette selection. The probability that an allocation will be selected by onlooker bee is calculated using Equation (10). The parameter fit_i is the fitness value of the solution. If an allocation is chosen by an onlooker bee, then it will be changed using Equation (9) as in the previous step.

$$P_i = \frac{fit_i}{\Sigma_{j=1}^{N} fit_j} \quad (10)$$

3) In this step (from line 14 to 15), we apply rotation and mutation. These quantum operators make it possible to approach the states of various allocations towards the state of the optimum one, and to disrupt a gene j by swapping positions of its α_j and β_j, respectively. The rotation [42] adjusts, i.e. increases or decreases, gene values of allocations according to the allocation with the lowest fitness value. The mutation [42] makes it possible to explore and exploit new solutions in the search space.
4) The last step (from line 20 to 23) takes place if an allocation is abandoned for a long period of time. Algorithmically speaking, this means that the fitness value of an allocation has not been improved for several generations. If this happens, scout bees generate a new solution using Equation (6) and replace the abandoned one.

D. Quality of found solutions

We discuss the quality of generated solutions. It is important to note that our algorithms use two optimization types, i.e. global and local. First, the global optimization is done by bidders using the firefly algorithm in order to select a task to perform – considering both Euclidean and Hamming distances (Eqs. (3) and)4)). So, if the parameters of this algorithm are well adjusted, then we can get very good results.

Second, the local optimization is done by the auctioneer combining artificial bee colony optimization – a numerical optimization method simulating the food-seeking behavior of honey bees – and quantum genetic algorithms – a heuristic optimization method simulating human evolutionary mechanisms and principles of quantum physics – in order to calculate an allocation for each task.

Numerical optimization by artificial bee colony has been widely used to successfully solve many real-world problems.

This method has a very good ability to explore the search space. However, in some cases its exploitation of the search space remains low and its convergence speed is also a problem. In order to overcome these limitations, this method has been combined with quantum genetic algorithms. QGA have the superposition principle, which greatly enriches diversity of solutions to be exploited and avoids premature convergence. In conclusion, the found allocations using this hybridization are quite promising, and this is shown in experimental section.

IV. Simulation and Result discussion

We evaluate the obtained results after simulating the proposed algorithms on some numerical data. We assume that all tasks are known beforehand – however, algorithms can also handle the case where tasks are progressively discovered (dynamic).

A. Generation of simulation data

The numerical data used for our simulations are randomly generated on a 2D grid of 100×100 cells. The number of used resources is 10. The speed, energy level, and aging factor of bidders are supposed to be constant for all.

These data have been divided into three datasets – i.e. dataset 1, dataset 2, and dataset 3. The first dataset has from 10 to 50 bidders and 100 tasks – bidders are incremented by 10. The second dataset has from 10 to 100 tasks and 10 bidders – tasks are incremented by 10. The third dataset has from 10 to 100 tasks and 50 bidders – tasks are incremented by 10. The purpose of dataset 1 is to observe the effect of the number of bidders on allocation performance, relative to the number of tasks. The purpose of datasets 2 and 3 is to observe the effect of the number of tasks on allocation performance, relative to the number of bidders. Finally, the data format used for bidders and tasks is exemplified in Tables III and IV.

B. Results and analyzes

In order to show the effectiveness of our solution, we compared its performance with the work presented in [11]. Both solutions were evaluated in terms of the allocation time and the number of allocated tasks. The names "FA-QABC-MRTA" and "FA-POWERSET-MRTA" are adopted to refer our solution and the solution proposed in [11], respectively.

The "FA-POWERSET-MRTA" [11] solution also uses two stages to address the MRTA problem. In the first stage, it behaves exactly like our solution, i.e. bidders use the firefly algorithm for tasks' choice. In the second stage, it calculates all subsets composed of all resources offered by bidders – for the considered task, then it chooses the one representing the optimal allocation. For the example depicted in Figure 2, the solution proposed by "FA-POWERSET-MRTA" must consider $2^{4+3+5+6}$ different subsets and choose the optimal allocation, i.e. the subset optimizing the cost function taken into account. Imagine its limit when the numbers of bidders and resources are large.

Tables V and VI show a comparison of "FA-QABC-MRTA" and "FA-POWERSET-MRTA" in terms of the allocation time and the utility values, respectively . We used the dataset 1 – 100 tasks and the number of bidders varies from 10 to 50. First, we notice that increasing the number of bidders does not greatly improve task allocation time for both solutions – i.e. on average, we have 20.199 seconds for "FA-POWERSET-MRTA" and 5.261 seconds for "FA-QABC-MRTA", no matter the number of bidders. So both solutions are able to allocate a large number of tasks with few bidders. However, we clearly see that "FA-QABC-MRTA" is significantly better than "FA-POWERSET-MRTA" in terms of allocation time – i.e. approximately 15 seconds of difference for each configuration. On the other hand, we also compared the utility values of these solutions. We notice that they are very close, that is to say, qualities of our allocations are close to qualities of optimal allocations.

Table V: Comparison of the "FA-QABC-MRTA" and "FA-POWERSET-MRTA" solutions for dataset 1 – allocation time (seconds).

Number of robots	FA–POWERSET–MRTA	FA–QABC–MRTA
Robots = 10	20,003	5,171
Robots = 20	20,019	5,234
Robots = 30	20,131	5,262
Robots = 40	20,372	5,298
Robots = 50	20,471	5,342

Table VI: Comparison of the "FA-QABC-MRTA" and "FA-POWERSET-MRTA" solutions for dataset 1 – utility values (unit).

Number of robots	FA–POWERSET–MRTA	FA–QABC–MRTA
Robots = 10	703,32	701,2
Robots = 20	798,79	795,47
Robots = 30	778,9	777,16
Robots = 40	723,08	720,79
Robots = 50	890,84	888,55

Figures 4 and 5 also show a comparison of "FA-QABC-MRTA" and "FA-POWERSET-MRTA" in terms of the allocation time. We used 10 bidders (dataset 2) and 50 bidders (dataset 3), respectively. The number of tasks varies from 10 to 100 in both cases. Again, we notice that increasing the number of bidders does not greatly improve the allocation time, no matter the number of tasks – with 10 or 50 bidders, we have approximately same times for both solutions. Naturally, the allocation time should increase as the number of tasks increases. However, we observe that its increase is almost linear. So, the solutions are effective in terms of temporal complexity. It should be noted that allocation time that is shown in these figures encompass all actions required for task allocation – i.e. communications between bidders and auctioneer, global allocation, and local allocation.

We graphically show the impact of the number of used bidders on the number of allocated tasks. Figure 6 shows the number of allocated tasks per cycle, according to the number of used bidders. We used 100 tasks and the number of bidders varies from 10 to 50. We notice that all tasks are allocated, no matter the used configuration. So, our solution

Table III: Example of bidders data

ID robot	Location	Resources	Costs	Velocity	Battery level	Aging factor
r_1	$(1, 1)$	$[1, 1, 1, 0]$	$[8, 6, 7, 0]$	10	1	1
r_2	$(9, 1)$	$[1, 1, 1, 1]$	$[8, 8, 7, 5]$	10	1	1
r_3	$(9, 5)$	$[0, 1, 1, 1]$	$[0, 6, 7, 9]$	10	1	1
r_4	$(1, 5)$	$[1, 1, 0, 1]$	$[8, 7, 0, 4]$	10	1	1

Table IV: Example of tasks data

ID task	Location	Resources	Duration
t_1	$(3, 3)$	$[1, 1, 1, 0]$	10
t_2	$(7, 3)$	$[1, 1, 0, 1]$	12

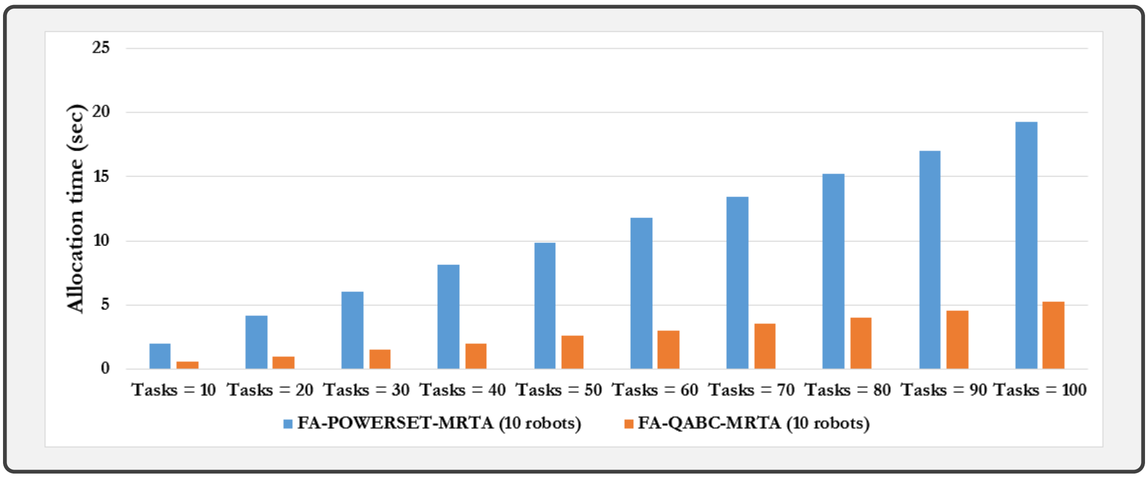

Figure 4: Comparison of the "FA-QABC-MRTA" and "FA-POWERSET-MRTA" solutions for dataset 2 – allocation time.

is efficient in terms of the number of allocated tasks. On the other hand, we confirm that increasing the number of bidders only increases the number of tasks allocated per iteration, but does not minimize the allocation time.

V. Conclusion and Perspectives

We proposed a solution to address the MRTA problem. The solution uses the auctions with two robot types: auctioneer (one instance) and bidders (several instances). In addition, the solution implements two optimization types: global (performed by bidders) and local (performed by the auctioneer). Our solution combined advantages of three well known heuristics. First, the firefly algorithm that distributes robots on tasks, i.e. each bidder chooses the task that it wants to execute. This step can be seen as a selection of tasks by bidders. In general the algorithms available in the literature do not frequently show this step, and usually perform it implicitly in the allocation process. The selection minimizes the number of bidders for each task, and it would minimize its allocation time. Secondly, artificial bee colony and quantum genetic algorithm are hybridized, due to their respective powers of exploration and exploitation, in order to allocate tasks to bidders. Simulation results showed that our proposed solution outperforms the work in [11] in terms of allocation time. Our solution solves the MRTA problem in a linear trend, and is efficient in terms of the number of allocated tasks – the rate of allocated tasks is 100%, no matter the adopted configuration. Finally, our solution supports cases where all tasks are known beforehand or discovered dynamically. As perspectives, we plan to further improve our solution by using energetic, temporal, and spatial constraints on tasks and robots. Also, we aim to implement our solution in real life problems (using real tasks and robots).

References

[1] A. Hussein, M. Adel, M. Bakr, O. M. Shehata, and A. Khamis, "Multi-robot task allocation for search and rescue missions," *Journal of Physics: Conference Series*, vol. 570, no. 5, p. 052006, 2014.

[2] A. Hussein and A. Khamis, "Market-based approach to multi-robot task allocation," in *International Conference on Individual and Collective Behaviors in Robotics (ICBR)*, 2013, Conference Proceedings, pp. 69–74.

[3] Y. Han, D. Li, J. Chen, X. Yang, Y. Hu, and G. Zhang, *A Multi-Robots Task Allocation Algorithm Based on Relevance and Ability With Group Collaboration*, 2010, vol. 3.

[4] A. Khamis, A. Hussein, and A. Elmogy, *Multi-robot Task Allocation: A Review of the State-of-the-Art*. Springer International Publishing, 2015, pp. 31–51.

[5] A. Orden, "The transhipment problem," *Manage. Sci.*, vol. 2, no. 3, pp. 276–285, apr 1956.

[6] H. W. Kuhn and B. Yaw, "The hungarian method for the assignment problem," *Naval Res. Logist. Quart*, pp. 83–97, 1955.

[7] B. P. Gerkey and M. J. Matari, "A formal analysis and taxonomy of task allocation in multi-robot systems," *The International Journal of Robotics Research*, vol. 23, no. 9, pp. 939–954, 2004.

[8] G. A. Korsah, A. Stentz, and M. B. Dias, "A comprehensive taxonomy for multi-robot task allocation," *The International Journal of Robotics Research*, vol. 32, no. 12, pp. 1495–1512, 2013.

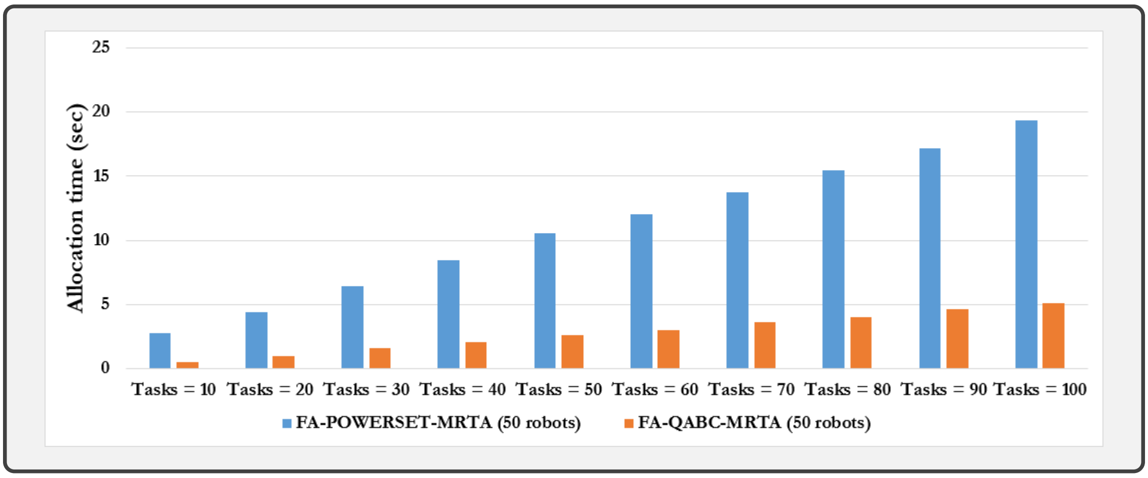

Figure 5: Comparison of the “FA-QABC-MRTA” and “FA-POWERSET-MRTA” solutions for dataset 3 – allocation time.

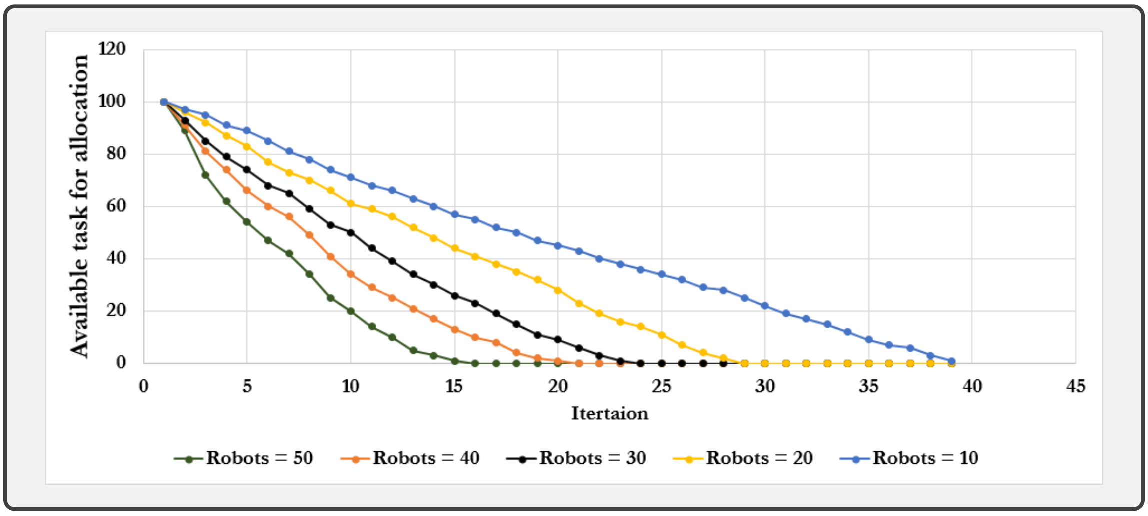

Figure 6: Number of allocated tasks according to the number of used bidders.

[9] E. Nunes, M. Manner, H. Mitiche, and M. Gini, “A taxonomy for task allocation problems with temporal and ordering constraints,” *Robot. Auton. Syst.*, vol. 90, no. C, pp. 55–70, April 2017.

[10] T. Sandholm, K. Larson, M. Andersson, O. Shehory, and F. Tohme, “Coalition structure generation with worst case guarantees,” *Artificial Intelligence*, January 1999.

[11] F. Zitouni and R. Maamri, “Fa-setpower-mrta: a solution for solving the multi-robot task allocation problem,” in *Computational Intelligence and Its Applications - 6th IFIP TC 5 International Conference, CIIA 2018, Oran, Algeria, May 8-10, 2018, Proceedings*, 2018, pp. 317–328.

[12] D. Vail and M. Veloso, “Multi-robot dynamic role assignment and coordination through shared potential fields,” in *Multi-Robot Systems*, L. P. A. Schultz and F. Schneider, Eds., January 2003.

[13] B. P. Gerkey and M. J. Mataric, “Sold!: auction methods for multirobot coordination,” *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 758–768, 2002.

[14] R. G. Simmons, D. Apfelbaum, W. Burgard, D. Fox, M. Moors, S. Thrun, and H. L. S. Younes, “Coordination for multi-robot exploration and mapping,” in *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*. AAAI Press, 2000, pp. 852–858.

[15] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, and P. H. Vance, “Branch-and-price: Column generation for solving huge integer programs,” *Operations Research*, vol. 46, pp. 316–329, 1996.

[16] J. Melvin, P. Keskinocak, S. Koenig, C. A. Tovey, and B. Y. Ozkaya, “Multi-robot routing with rewards and disjoint time windows.” in *IROS*, 2007, pp. 2332–2337.

[17] M. B. Dias, “Traderbots: A new paradigm for robust and efficient multirobot coordination in dynamic environments,” Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, PA, January 2004.

[18] M. Berhault, H. Huang, P. Keskinocak, S. Koenig, W. Elmaghraby, P. Griffin, and A. Kleywegt, “Robot exploration with combinatorial auctions,” in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, vol. 2, 2003, Conference Proceedings, pp. 1957–1962.

[19] S. Koenig, C. Tovey, X. Zheng, and I. Sungur, “Sequential bundle-bid single-sale auction algorithms for decentralized control,” in *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, ser. IJCAI’07. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2007, pp. 1359–1365.

[20] M. G. Lagoudakis, E. Markakis, D. Kempe, P. Keskinocak, A. Kleywegt, S. Koenig, C. Tovey, A. Meyerson, and S. Jain, “Auction-based multi-robot routing,” in *Proceedings of Robotics: Science and Systems*, Cambridge, USA, June 2005.

[21] S. C. Botelho and R. Alami, “M+: a scheme for multi-robot cooperation through negotiated task allocation and achievement,” in *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, vol. 2, 1999, Conference Proceedings, pp. 1234–

1239.
[22] D. C. Mackenzie, "Collaborative tasking of tightly constrained multi-robot missions," in *In Multi-Robot Systems: From Swarms to Intelligent Automata: Proceedings of the 2003 International Workshop on Multi-Robot Systems*. Kluwer Academic Publishers, 2003.
[23] S. Chien, A. Barrett, T. Estlin, and G. Rabideau, "A comparison of coordinated planning methods for cooperating rovers," in *Proceedings of the Fourth International Conference on Autonomous Agents*, ser. AGENTS '00. New York, NY, USA: ACM, 2000, pp. 100–101.
[24] F. Zitouni and R. Maamri, "Cooperative learning-agents for task allocation problem," in *Interactive Mobile Communication Technologies and Learning*, M. E. Auer and T. Tsiatsos, Eds. Cham: Springer International Publishing, 2018, pp. 952–968.
[25] T. Lemaire, R. Alami, and S. Lacroix, "A distributed tasks allocation scheme in multi-uav context," in *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, vol. 4, 2004, Conference Proceedings, pp. 3622–3627.
[26] O. Shehory and S. Kraus, "Task allocation via coalition formation among autonomous agents," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 1*, ser. IJCAI'95. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1995, pp. 655–661.
[27] ——, "Methods for task allocation via agent coalition formation," *Artificial Intelligence*, vol. 101, no. 1, pp. 165–200, 1998.
[28] L. Vig and J. A. Adams, "Multi-robot coalition formation," *IEEE Transactions on Robotics*, vol. 22, no. 4, pp. 637–649, 2006.
[29] J. Guerrero and G. Oliver, "Multi-robot task allocation strategies using auction-like mechanisms," *Artificial Research and Development in Frontiers in Artificial Intelligence and Applications*, vol. 100, pp. 111–122, 2003.
[30] L. Liu and Z. Zhiqiang, "Combinatorial bids based multi-robot task allocation method," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, 2005, Conference Proceedings, pp. 1145–1150.
[31] P. M. Shiroma and M. F. M. Campos, "Comutar: A framework for multi-robot coordination and task allocation," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009, Conference Proceedings, pp. 4817–4824.
[32] M. Koes, I. Nourbakhsh, and K. Sycara, "Constraint optimization coordination architecture for search and rescue robotics," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, 2006, Conference Proceedings, pp. 3977–3982.
[33] G. A. Korsah, "Exploring bounded optimal coordination for heterogeneous teams with cross-schedule dependencies," Ph.D. dissertation, Carnegie Mellon University, Pittsburgh, PA, January 2011.
[34] E. G. Jones, M. B. Dias, and A. Stentz, "Time-extended multi-robot coordination for domains withintra-path constraints," *Autonomous Robots*, vol. 30, no. 1, pp. 41–56, Jan 2011.
[35] L. E. Parker and T. Fang, "Building multirobot coalitions through automated task solution synthesis," *Proceedings of the IEEE*, vol. 94, no. 7, pp. 1289–1305, 2006.
[36] R. M. Zlot, "An auction-based approach to complex task allocation for multirobot teams," Ph.D. dissertation, Pittsburgh, PA, USA, 2006, aAI3250901.
[37] F. Tang and L. E. Parker, "A complete methodology for generating multi-robot task solutions using asymtre-d and market-based task allocation," in *Proceedings 2007 IEEE International Conference on Robotics and Automation*, 2007, Conference Proceedings, pp. 3351–3358.
[38] R. G. Smith, "The contract net protocol: High-level communication and control in a distributed problem solver," *IEEE Transactions on Computers*, vol. C-29, no. 12, pp. 1104–1113, 1980.
[39] X.-S. Yang, *Nature-Inspired Metaheuristic Algorithms*. Luniver Press, 2008.
[40] ——, "Firefly algorithms for multimodal optimization," in *Stochastic Algorithms: Foundations and Applications*, O. Watanabe and T. Zeugmann, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 169–178.
[41] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (abc) algorithm," *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, Nov 2007.
[42] R. Lahoz-Beltra, "Quantum genetic algorithms for computer scientists," *Computers*, vol. 5, no. 4, p. 24, 2016.