Swarm and Evolutionary Computation 44 (2019) 480–495

Contents lists available at ScienceDirect

Swarm and Evolutionary Computation

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

Weighted strategies to guide a multi-objective evolutionary algorithm for multi-UAV mission planning

Cristian Ramirez Atencia [a], Javier Del Ser [b,c,d], David Camacho [a,*]

[a] Departamento de Ingeniería Informática, Universidad Autónoma de Madrid, C/Francisco Tomás y Valiente 11, 28049 Madrid, Spain
[b] OPTIMA Area, TECNALIA, E-48160 Zamudio, Spain
[c] Department of Communications Engineering, University of the Basque Country (UPV/EHU), E-48013 Bilbao, Spain
[d] Basque Center for Applied Mathematics (BCAM), E-48009 Bilbao, Spain

ARTICLE INFO

Keywords:
Unmanned air vehicle
Mission planning
Constraint satisfaction problem
Multi-objective evolutionary algorithm
Weighted strategies

ABSTRACT

Management and mission planning over a swarm of unmanned aerial vehicle (UAV) remains to date as a challenging research trend in what regards to this particular type of aircrafts. These vehicles are controlled by a number of ground control station (GCS), from which they are commanded to cooperatively perform different tasks in specific geographic areas of interest. Mathematically the problem of coordinating and assigning tasks to a swarm of UAV can be modeled as a constraint satisfaction problem, whose complexity and multiple conflicting criteria has hitherto motivated the adoption of multi-objective solvers such as multi-objective evolutionary algorithm (MOEA). The encoding approach consists of different alleles representing the decision variables, whereas the fitness function checks that all constraints are fulfilled, minimizing the optimization criteria of the problem. In problems of high complexity involving several tasks, UAV and GCS, where the space of search is huge compared to the space of valid solutions, the convergence rate of the algorithm increases significantly. To overcome this issue, this work proposes a weighted random generator for the creation and mutation of new individuals. The main objective of this work is to reduce the convergence rate of the MOEA solver for multi-UAV mission planning using weighted random strategies that focus the search on potentially better regions of the solution space. Extensive experimental results over a diverse range of scenarios evince the benefits of the proposed approach, which notably improves this convergence rate with respect to a naïve MOEA approach.

1. Introduction

Unmanned aircraft system(UASs), also referred to as drones, unmanned aerial vehicle (UAV), and remotely piloted aircraft system (RPAS), can be defined as an aircraft without a human pilot on-board. Instead, the UAS is controlled from an operator on the ground. The main difference between UAV and RPAS resides on the fact that UAV are basically planes that do not have a pilot, and no human intervention is made during the flight plan. UAV rely on a pre-programmed flight plan, and it is the on-board computer of the aircraft what actually controls it and reacts to changing conditions in order to reach the desired location. However, in case of RPAS, the need for a pilot is not really overridden but the pilot is just relocated to a remote location. The RPAS still operates as a typical airplane: the pilot controls it from a cockpit with all the necessary controls, and the inputs provided by the pilot are transmitted to the remote remotely piloted aircraft (RPA). This paper will focus on UAV, so the aircraft control will rely on a pre-programmed flight plan.

The advent of pilot-less aerial vehicles and rapid development of their capabilities have paved the way towards new military and commercial applications. UAV have become very popular in many applications including surveillance [1], training [2], disaster and crisis management [3] and firefighting [4], since they avoid risking human lives while their manageability permits to reach areas of hard access. Possibilities for UAV span further towards the last-mile delivery of goods [5] and support for wireless communications [6,7], among many others (see Ref. [8] for a comprehensive survey).

The process of mission planning for a collaborative swarm of UAV involves generating tactical goals, commanding vehicles, risk avoidance, coordination and timing. Currently, UAV are controlled remotely by human operators from GCS, using rudimentary planning systems, such as pre-configured plans, classical planners that are not able to

* Corresponding author.
E-mail addresses: cristian.ramirez@inv.uam.es (C. Ramirez Atencia), javier.delser@tecnalia.com (J. Del Ser), david.camacho@uam.es (D. Camacho).

https://doi.org/10.1016/j.swevo.2018.06.005
Received 11 August 2017; Received in revised form 27 March 2018; Accepted 3 June 2018
Available online 13 June 2018
2210-6502/© 2018 Elsevier B.V. All rights reserved.

cope with the entire complexity of the problem, or manually provided schedules. Classical planners are based on a graph search or resort to a logic engine. Remarkably, this kind of planners usually undergo severe practical limitations, which add to the high computational resources demanded by their underlying solvers [9]. When undertaking complex coordinated missions, planning systems demand more efficient problem-solving capabilities to cope with conflicting objectives and stringent constraints over both spatial and time domains. Therefore, the so-called mission planning problem (MCMPP) have lately gained momentum in the research community, propelled by a notable increase in the scales and number of applications foreseen for UAV in the near future.

Nowadays most GCS consider a relation between N operators and 1 UAV due to the high complexity of a mission. The purpose of this work is to present a novel approach to reduce this complexity, so this relation can be swapped, first to one between N operators and UAV, and in a near future to a single operator controlling multiple vehicles.

In the literature several algorithmic approaches have been explored and reported for UAV mission planning. Fabiani et al. [10] modelled the problem for search and rescue scenarios using markov decision process (MDP) and solve it with dynamic programming algorithms. Leary et al. [11] compared the performance of mixed-integer linear programming (MILP), simulated annealing (SA) 49 and auction algorithm (AA) for mission planning problems with multiple UAV. Wang et al. [12] represented the task scheduling of UAV as a MILP and used tabu search (TS) to solve this problem. Shang et al. [13] proposed a combination of genetic algorithm (GA) and ant colony optimization (ACO) for solving the UAV team oriented problems.

An essential concept in mission planning is cooperation or collaboration, which occurs at a higher level when various UAV work together in a common mission sharing data and controlling actions together. There are few contributions that deal with Multi-UAV problems in a deliberative paradigm (cooperative task assignment and mission planning). Bethke et al. [14] proposed an algorithm for cooperative task assignment that extends the receding-horizon task assignment (RHTA) algorithm to select the optimal sequence of tasks for each UAS. Another approach by Kvarnstrom et al. [15] proposes a new mission planning algorithm for collaborative UAV based on combining ideas from forward-chaining planning with partial-order planning. This approach led to a new hybrid partial-order forward-chaining (POFC) framework that meets the requirements on centralization, abstraction, and distribution found in realistic emergency services settings. Other works have been focused on distributed approaches for solving collaborative mission planning. Pascarell et al. [16] proposed a core paths graph (CPG) algorithm for trajectory planning. Finally, other approaches formulate the mission planning problem as a constraint satisfaction problem (CSP) [17], where the tactic mission is modelled and solved using CSP techniques [18], in combination with genetic algorithms [19] or represented inside a distributed multi-agent system [20,21].

Most of the above contributions assume that UAV are commanded by a single GCS, but in practice multi-UAV missions require the use of several GCS for controlling all the UAV involved in the mission scenario. This multi-GCS approach makes the underlying problem even more complex. Besides, several criteria can be adopted to quantify the quality of a solution, such as the fuel consumption, the makespan, the cost of the mission, or the number of UAV or GCS to employ and different risk factors that could compromise the mission, among others. In these cases, the problem is considered as a multi-objective optimization problem (MOP), for which an estimation of the pareto optimal frontier (POF) must be inferred so as to get a portfolio of solutions (mission plans) differently albeit optimally balancing the considered conflicting objectives. To this end, an algorithmic option is to use MOEA, which are inspired from evolutionary reproduction and mutation mechanisms to efficiently construct an estimation of the POF for a given MOP. The literature is scarce in what refers to this family of multi-criteria solvers when applied to multi-GCS MCMPP, with references dealing with path planning rather than mission scheduling (see e.g. Ref. [22] and references therein).

The vehicles considered in this work are based on Airbus Defence & Space UAV,[1] as part of the SAVIER project. Four types of UAV have been considered: medium altitude long endurance (MALE), high altitude long endurance (HALE), unmanned combat aerial vehicle (UCAV) and unmanned reconnaissance aerial vehicle (URAV), this last one is based on the EADS ATLANTE model.[2] All utilized vehicles have 6 degrees of freedom (DoF). In addition, the definition of the missions, the flight profiles and the following procedures have been specified by Airbus engineers involved in this project according to realistic standards in aerial simulations.

This manuscript extends and improves a previous approach dealing with a MOEA to tackle this family of problems [19]. In this new approach, the encoding consists of different alleles representing the features of the problem, whereas the fitness function builds upon several conflicting objectives of the application scenario, which are to be minimized subject to a number of constraints. Unfortunately, in some problems of high complexity involving several tasks, UAV and GCS, where the space of search is huge compared to the space of valid solutions, this approach is unable to converge in a reasonable time. Therefore, in order to improve it and guide the algorithm to valid solutions, this work takes a step further by proposing a novel *weighted random generator* for the formation of new individuals. This generator is applied in three parts of the encoding:

- First, we use a weighted random function that assigns lower probability to individuals with higher numbers of UAV for performing the tasks.
- Second, the selection of the UAV(s) performing the task at hand is done by using a weighted random function driven by the distance of the UAV(s) to the task.
- Third, GCS controlling a UAV is selected by means of a weighted random function depending on the distance of the GCS to the UAV.

To test this approach, several weighted strategies are proposed and assessed over a number of realistic scenarios with varying scales. The obtained results are promising and encourage the adoption of the proposed simple heuristics in MOEA applied to other multi-criteria mission planning scenarios.

The rest of the paper is structured as follows. Section 2 provides a baseline mathematical formulation of the considered UAV mission planning problem, including an overview of the CSP modelling of the problem. Section 3 presents the designed solver to efficiently tackle the aforementioned problem, providing details on its encoding, constraint handling strategy and the weighted strategies used for producing and evolving individuals. Section 4 discusses the results obtained from several computer experiments using realistic data. Finally, the last section draws conclusions and outlines future research lines related to this work.

2. System model and problem formulation

The MCMPP can be defined as a $\mathtt{T}$-sized set of *tasks* $\mathcal{T} \doteq \{T_1, T_2, \ldots, T_{\mathtt{T}}\}$ to be performed by a swarm of $\mathtt{U}$ UAV $\mathcal{U} = \{U_1, U_2, \ldots, U_{\mathtt{U}}\}$ within a specific time interval. Each mission should be performed in a specific geographic zone. In addition, G GCS $\mathcal{G} \doteq \{G_1, G_2, \ldots, G_{\mathtt{G}}\}$ control the swarm of UAV. A basic mission planning comprises the assignment of each task T_t (with $t \in \{1, \ldots, \mathtt{T}\}$) to a specific UAV U_u (corr. $u \in \{1, \ldots, \mathtt{U}\}$), as well as each UAV to a specific GCS G_g ($g \in \{1, \ldots, \mathtt{G}\}$), ensuring that the mission can be successfully performed within a time frame. Such a solution can be mathematically formulated as two many-to-one mapping functions

[1] http://www.airbus.com/defence/uav.html.
[2] http://www.airforce-technology.com/projects/atlante-uav/.

$\lambda_{TU} : \{1, \dots, \mathtt{T}\} \mapsto \{1, \dots, \mathtt{U}\}$ and $\lambda_{UG} : \{1, \dots, \mathtt{U}\} \mapsto \{1, \dots, \mathtt{G}\}$ such that $\lambda_{TU}(t) = u$ and $\lambda_{UG}(u) = g$ denotes that task T_t is assigned to UAV U_u, which in turn is commanded by GCS G_g.

The set $\mathcal{T}$ can collect very diverse tasks, from photographing to escorting a target, monitoring a zone or providing support for on-site communications. Some of such tasks can be performed by several UAV (Multi-UAV), hence reducing the time needed to accomplish them, e.g. mapping an area or the so-called step & stare. Each task must be performed over a particular geographic area and within a specific time interval. Furthermore, tasks require the use of sensors installed on every UAV (i.e. electro-optical or infra-red (EO/IR) cameras, synthetic aperture radar (SAR), inverse synthetic aperture radar (ISAR) maritime patrol radar (MPR), etc.) participating in the mission. To perform a specific task, the UAV(s) assigned to perform it must use one specific sensor while on operation. Although in real scenarios, some complex task may precise the use of several sensors of the same vehicle at the same time, this special case has been omitted in this problem. This poses further constraints to the mission planning, as it might be the case that not all UAV in the swarm have all type of sensors installed on board, hence not being eligible to undertake any task within $\mathcal{T}$. This being said, UAV performing the mission have some features that must be taken into account in order to check if a mission plan is correct:

- The initial position, in terms of latitude (LAT) and longitude (LONG), of the UAV U_u (with $u \in \{1, \dots, \mathtt{U}\}$), denoted as $\mathrm{Pos}_u(\tau) \doteq (\mathrm{LAT}_u(\tau), \mathrm{LON}_u(\tau))$, with τ denoting time.
- The fuel of the UAV during the mission plan, denoted as $\mathrm{FUEL}_u(t)$ (in kilograms).
- The cost per hour of usage of a UAV C_u (in monetary units per hour).
- The maximum flight time and range of UAV U_u, measured as the $T_u^{\max}$ and $D_u^{\max}$.
- The available sensors at every UAV, by which a binary correspondence $\xi : \{1, \dots, \mathtt{U}\} \times \{1, \dots, \mathtt{T}\} \mapsto \{0, 1\}$ can be built such that $\xi(u, t) = 1$ if UAV $U_u \in \mathcal{U}$ has all sensors required to perform task $T_t \in \mathcal{T}$ installed on board, and $\xi(u, t) = 0$ otherwise. Clearly, $\lambda_{TU}(t, u) = 0$ whenever $\xi(u, t) = 0$ (i.e. no UAV should undertake any task for which it is unqualified). When there is more than a sensor on-board capable of performing a given mission, the selection of which sensor to use should be included as another design criterion for the plan due to the direct involvement of this variable in the cost and the maximum flight time of the UAV. Sensors can be non-cooperative (such as the cameras used in a target photographing task) or cooperative (correspondingly, radars working together in a surveillance task). These cooperative sensors are used in Multi-UAV tasks, where different vehicles cooperate and their sensors coordinate among themselves to achieve the best performance when undertaking the specific task.
- $\mathtt{F} \geq 1$ flight profiles $\mathcal{F} \doteq \{F_1, \dots, F_{\mathtt{F}}\}$, each specifying speed $S_f(\tau)$, fuel consumption rate in kilograms per hour $R_f^{\mathrm{FUEL}}(\tau)$ and flight altitude $A_f(\tau)$ in feet (or equivalently, ascent/descent angle) during the different time instants τ of the profile. Since these operational parameters have direct implications on most of the above features (e.g. fuel consumption of the UAV, its maximum flight time and the range or maximum flight distance), flight profiles are also optimization variables to be assigned so as to complete the mission plan successfully. In this problem, four flight profiles have been considered for each UAV: a climb profile, used for every altitude rise; a descent profile, used for every descent of altitude; a minimum consumption profile, where the speed is low and the altitude is high, so the fuel consumption is at its minimum; and a maximum speed profile, where the speed is at its maximum value, the altitude low, and so the fuel consumption is high.

Additionally, there could exist vehicle and time dependencies between different tasks in a mission. Vehicle dependencies reflect whether two tasks can be assigned to the same UAV or instead must be mapped to different UAV, i.e. whether the task-to-UAV mapping function $\lambda_{TU}(t)$ as defined above is not injective. Time dependencies pose restrictions on the start and end times of related tasks so they are performed simultaneously, consecutively or in any other mutual relation of dependency along time. Such temporal relations can be expressed in terms of the Allen's interval algebra [23] as follows:

- $T_t < T_{t'}$: T_t takes place before $T_{t'}$.
- $T_t \, m \, T_{t'}$: T_t *meets* $T_{t'}$ (i.e. $T_{t'}$ starts right after T_t).
- $T_t \, o \, T_{t'}$: T_t overlaps $T_{t'}$.
- $T_t \, s \, T_{t'}$: T_t starts jointly with $T_{t'}$.
- $T_t \, d \, T_{t'}$: T_t starts and ends while $T_{t'}$ is performed.
- $T_t \, f \, T_{t'}$: T_t finishes together with $T_{t'}$.
- $T_t = T_{t'}$: T_t is equal to $T_{t'}$.

Similarly, the set of GCS controlling the deployed UAV also feature several aspects to be considered when checking a mission plan:

- The geographical position where each GCS is located, defined as $\mathrm{Pos}_g \doteq (\mathrm{LAT}_g, \mathrm{LON}_g)$ for $g \in \{1, \dots, \mathtt{G}\}$.
- The maximum number of UAV that every GCS G_g can control, given by $1 \leq \mathtt{U}_g^{\max} \leq \mathtt{U}$.
- The permitted types of UAV that every GCS can command.
- The coverage or within range of the communications equipment of every GCS, which is assumed to be circularly shaped with radius $R_g^{\max}$ (in nautical miles).

Fig. 1 presents a mission scenario with $\mathtt{T} = 16$ tasks (represented as green squares and lines, and symbols as photo or tracking that denote a specific task to be done in that area), $\mathtt{U} = 8$ UAV and $\mathtt{G} = 3$ GCS. As shown in this figure, the zone of the mission contains several no flight zone (NFZ) shaded in red. These zones must be avoided in the planned trajectories of the UAV during the mission, impacting on the duration of routes traced over the scenario and ultimately, on the fuel consumption, maximum flight time, range and other parameters related to the dynamics of the application scenario.

When a task is assigned to a vehicle, it is necessary to compute the total time taken for the UAV to complete the path between its departure location and the area where the task is to be performed. If a task T_t is the last one assigned to a vehicle U_u, the time taken to return from this last task to the base must be also calculated. In order to compute these times, it is necessary to know which of the set of UAV's flight profiles $\mathcal{F}$ will be used, providing the fuel consumption ratio, speed and altitude as previously mentioned.

Furthermore, the assignment of tasks to UAV is also subject to the order in which such tasks are accomplished during the mission plan, always in compliance with the time constraints between tasks. In mathematical terms we will define the mapping function $\Gamma_{TU} : \{1, \dots, \mathtt{T}\} \times \{1, \dots, \mathtt{U}\} \mapsto \{0, \dots, \mathtt{T} - 1\}$, which establishes an order for every UAV $U_u \in \mathcal{U}$ to accomplish any given task $T_t \in \mathcal{T}$. It should be clear that $\Gamma_{TU}(t, u)$ exists iff (*if and only if*) $\lambda_{TU}(t, u) = 1$, i.e. UAV U_u is assigned to perform task T_t. For UAV $U_u \in \mathcal{U}$ the subset of its assigned tasks will be given by $\Gamma_{U_u} = \{\Gamma_{TU}(t, u) : \lambda_{TU}(t, u) = 1\}$, such that $\cup_{u=1}^{V} \Gamma_{U_u} = \mathrm{Perm}(|\mathcal{T}|)$, where $\mathrm{Perm}(a)$ denotes a permutation of the integer set $\{0 \dots, a - 1\}$. In addition, these orders must satisfy the time constraints between any pair of tasks T_t and $T_{t'}$ (with $t \neq t'$). For example, if UAV U_1 performs tasks $T_1 \to T_5$, we could define $\Gamma_{U_1} = \{4, 5, 1, 3, 2\}$ such that $T_5 < T_1$ and $T_3 < T_2$.

Given the above definitions, a mission plan $\mathbf{\Phi}$ can be defined as

$$\mathbf{\Phi} \doteq [\lambda_{TU}, \Gamma_{TU}, \lambda_{UG}, \lambda_{TUF}, \lambda_{TUS}], \tag{1}$$

where $\lambda_{TUF} : \{1, \dots, \mathtt{T} + 1\} \times \{1, \dots, \mathtt{U}\} \mapsto \{1, \dots, \mathtt{F}\}$ is a mapping function that establishes flight profiles to any UAV/task combination (including task $T_{\mathtt{T}+1} = \mathtt{ReturnToBase}$); and $\lambda_{TUS} : \{1, \dots, \mathtt{T}\} \times \{1, \dots, \mathtt{U}\} \mapsto \{1, \dots, \mathtt{S}\}$ similarly indicates the sensor selection (if any) for UAV $U_u \in \mathcal{U}$ to accomplish any given task $T_t \in \mathcal{T}$. It should be clear that $\lambda_{TUS}(t, u)$ exists iff (*if and only if*) $\xi(u, t) = 1$, i.e.

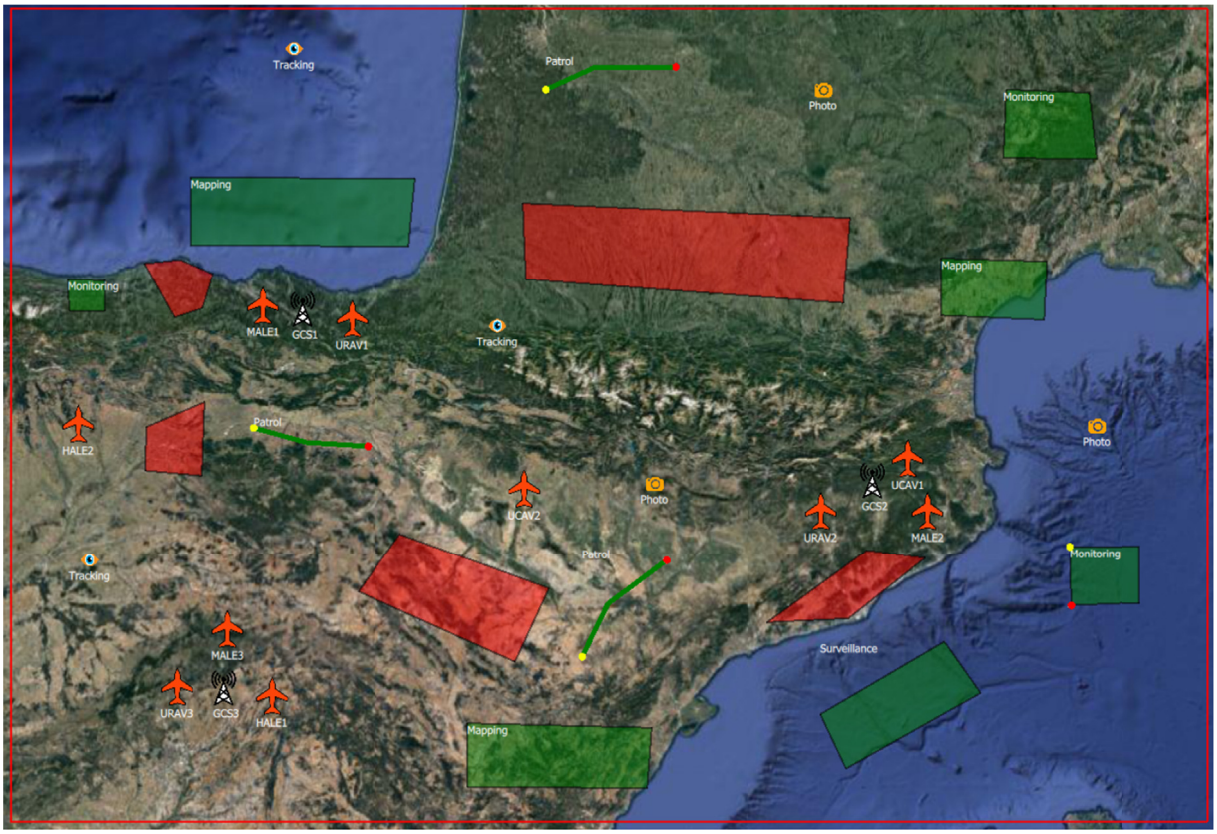

Fig. 1. Snapshot exemplifying a mission with $\mathtt{T} = 16$ tasks (4 of them comprising multiple UAV), $\mathtt{U} = 8$ UAV and $\mathtt{G} = 3$ GCS.

UAV U_u is qualified to undertake task T_t.

The quality of a given mission plan $\mathbf{\Phi}$ is driven by the Pareto optimality of a set of conflicting objectives to be simultaneously optimized:

1. The total cost of the mission $C^T(\mathbf{\Phi})$, given by $C^T(\mathbf{\Phi}) = \sum_{u=1}^{\mathtt{U}} C_u T_u(\mathbf{\Phi})$, where $T_u(\mathbf{\Phi})$ is the total flight time of UAV u along its routes in plan $\mathbf{\Phi}$.
2. The end time of the mission or makespan $T^{\max}(\mathbf{\Phi})$, i.e. the time at which the last UAV $u \in \mathcal{U}$ returns to the base.
3. The risk of the mission $\delta(\mathbf{\Phi}) \in \mathbb{R}[0, 100]$, which is computed as an average percentage indicating how risky the mission is. We consider four risk factors in this problem: UAV that finish the mission with *low fuel* (with respect to a given threshold FUEL^{th}), UAV that *fly near to the ground* (depending on the route and the altitude $A_f(\tau)$ of the adopted flight profile through the choice of λ_{TUF}), UAV that *fly out of the coverage or line of sight (LOS) of the GCS* controlling them; and UAV that *fly close between them*, which intuitively depends on the time constraints between concurrently performed tasks and eventual spatial overlaps among routes/flight profiles. The risk considered in the mission relates to safety by means of factors whose values are enforced to lie within an admissible numerical range that does not interfere with the performance of the mission.
4. The number of UAV $\mathtt{U}^{eff} \leq \mathtt{U}$ used in the mission plan, given by $\mathtt{U}^{eff}(\mathbf{\Phi}) \doteq \sum_{u=1}^{\mathtt{U}} \mathbb{I}(T_u(\mathbf{\Phi}) > 0)$, where $\mathbb{I}(\cdot)$ is a binary function taking value 1 if its argument holds (and 0 otherwise).
5. The total fuel consumption $\text{FUEL}(\mathbf{\Phi})$, given by the difference between the initial fuel load $\sum_{u=1}^{\mathtt{U}} \text{FUEL}_u(0)$ in the UAV prior to the start of the mission and that remaining $\sum_{u=1}^{\mathtt{U}} \text{FUEL}_u(T^{\max}(\mathbf{\Phi}))$ once the mission is over.
6. The total flight time $T(\mathbf{\Phi})$, given by $\sum_{u=1}^{\mathtt{U}} T_u(\mathbf{\Phi})$.
7. The total distance $D(\mathbf{\Phi})$ traversed, given by $\sum_{u=1}^{\mathtt{U}} D_u(\mathbf{\Phi})$, where $D_u(\mathbf{\Phi})$ is the total distance traversed by UAV u along its routes in mission plan $\mathbf{\Phi}$.

Before proceeding with the formal problem statement, some intuition must be given on the confluence of the above criteria in a many-objective design problem. On one hand, using many UAV is beneficial for minimizing the total distance traversed, the makespan of the mission, yet incurs a penalty in terms of cost. On the other hand, opting for a few UAV would yield reduced costs for the mission, but at the cost of increasing the risk of the mission. Likewise, the interplay between a moderate number of selected UAV with the above fitness measures is unclear and stringently subject to a proper choice of the number of sensors, the order in which tasks are performed and the number of UAV assigned to the task at hand, among many other decisions. Therefore, it is crucial to pose all these metrics in a many-objective optimization problem so as to assess their Pareto-optimal interactions.

The MCMPP problem can be then formulated as follows:

$$\underset{\mathbf{\Phi}}{\text{Min}} \quad C^T(\mathbf{\Phi}), T^{\max}(\mathbf{\Phi}), \delta(\mathbf{\Phi}), \mathtt{U}^{eff}(\mathbf{\Phi}), \text{FUEL}(\mathbf{\Phi}), T(\mathbf{\Phi}), D(\mathbf{\Phi}), \tag{2}$$

$$\text{s. t.} \quad \lambda_{TUS}(t, u) = s \text{ iff } \xi(u, t) = 1 \text{ and } \lambda_{TU}(t) = u, \tag{3}$$

$$\sum_{u=1}^{\mathtt{U}} \mathbb{I}(\lambda_{UG}(u) = g) \leq U_g^{\max}, \ \forall g \in \mathcal{G}, \tag{4}$$

$$\|Pos_u(\tau) - Pos_g\|_F < R_g^{\max} \text{ if } \lambda_{UG}(u) = g, \ \forall \tau \in [0, T^{\max}(\mathbf{\Phi})], \tag{5}$$

$$f : \lambda_{TUF}(t, u) = f \text{ if } \lambda_{TU}(t) = u \text{ are feasible}, \tag{6}$$

$$A_f(\tau) \geq 0 \ \forall \tau \text{ and } \forall f \in \mathcal{F}\text{: } \lambda_{TUF}(t, u) = f \text{ if } \lambda_{TU}(t) = u, \tag{7}$$

$$\|Pos_{u_1}(\tau) - Pos_{u_2}(\tau)\|_F > 0, \ \forall u_1 \neq u_2 \in \mathcal{U}, \ \forall \tau \in [0, T^{\max}(\mathbf{\Phi})], \tag{8}$$

$$\text{FUEL}_u(\tau) > 0, \ \forall \tau \in [0, T^{\max}(\mathbf{\Phi})] \text{ if } \lambda_{TU}(t) = u \text{ for some } t \in \mathcal{T}, \tag{9}$$

$$T_u(\mathbf{\Phi}) \leq T_u^{\max},\ \forall u \in \mathcal{U}, \tag{10}$$

$$D_u(\mathbf{\Phi}) \leq D_u^{\max},\ \forall u \in \mathcal{U}, \tag{11}$$

$$\mathbf{\Phi} \text{ (through } \lambda_{TU}(\cdot) \text{ and } \Gamma_{TU}) \text{ fulfills time and vehicle constraints} \tag{12}$$

where Expression (3) denotes sensor constraints, i.e. a UAV assigned to a task should have the sensors needed to perform it installed on board; inequality (4) constraints the maximum number of UAV allowed by a GCS; the inequality (5) stands for a guaranteed coverage of every UAV to the GCS to which it is allocated; the notion of feasibility in Expression (6) ensure that the UAV perform their paths for every task and in its return to the base without any incident (as will be later detailed, a Theta* algorithm [24] is used to assure that the vehicle avoids the NFZ, and that every climb/descent needed for the change of altitude is attainable); a positive distance to the ground, between different UAV and fuel loaded at the UAV at every time τ are reflected in inequalities (7), (8) and (9), respectively; Expressions (10) and (11) assure that the flight time and distance traversed, respectively, for each UAV do not overpass their maximum flight time and range; and time and vehicle dependency constraints are accounted in Expression (12).

2.1. CSP modeling of the mission planning problem

One alternative to model the above MCMPP problem is by resorting to CSP, since we seek the optimal schedule of resource-task assignments satisfying a set of imposed constraints. Thereby, we define:

- A set of variables $\mathcal{X} = \{x_1, \dots, x_L\}$.
- For each variable, a finite set D_l of possible values (its domain).
- And a set of constraints $\mathcal{C} = \{C_1, \dots, C_K\}$ restricting the values that variables can simultaneously take.

A solution to a CSP is an assignment of a value $d_l \in D_l$ to $x_l \in \mathcal{X}$ that satisfies all the imposed constraints. If all the variables have a value, then we deal with a complete instantiation of the variables. Otherwise, the variables are *partially* instantiated. If we assign a value to an uninstantiated variable in a partial instantiation, then we *extend* the partial instantiation. We will refer to an instantiation of a variable as $< x, v >$. The projection of a constraint C_k on the variables $\{x_1, \dots, x_{L'}\}$, denoted as $P(C_k, \{x_1, \dots, x_{L'}\})$, is the set of all $\{v_1, \dots, v_{L'}\}$ values for which it holds that the partial instantiation $< x_1, v_1 >, \dots, < x_{L'}, v_{L'} >$ can be extended to hold the constraints. When we add time to the problem, then we have a temporal constraint satisfaction problem (TCSP). It is a particular class of CSP where variables represent times (time points, time intervals or durations) and constraints establish allowed temporal relations between them [25].

The most widely used representation of a CSP is a graph where the pairs <Variable, Value> are the nodes and the constraints are the edges. The main techniques to solve CSP are:

- Search algorithms: cross the space of partial instantiations, building a complete instantiation that satisfies all the constraints, or determine that the problem is inconsistent.
- Propagation algorithms: infer new constraints that are added to the problem without loosing any information. The worst case time complexity is polynomial in the size of the problem. They are fast in discovering local inconsistencies.
- Structure-driven algorithms: is a mix of the two above techniques.

The MCMPP is modelled as a CSP, where a graph representation of the problem is presented in Fig. 2. For this problem, two types of variable are used: the variables of the mission plan (represented in blue in the graph), given by the previous definition of $\mathbf{\Phi}$; and several auxiliary variables (represented in green in the graph) that are computed during the CSP propagation process and employed to ease the formulation of the constraints of the problem, especially those referred to time points and intervals. The different constraints of the problem are represented in yellow boxes, and the optimization objectives are represented in red ovals.

The variables of the CSP are then:

- *Task Assign*: this variable corresponds to λ_{TU} in the MCMPP.
- *Orders*: this variable corresponds to Γ_{TU} in the MCMPP.
- *GCS Assign*: this variable corresponds to λ_{UG} in the MCMPP.
- *Sensor Assign*: this variable corresponds to λ_{TUS} in the MCMPP.
- *Path FP*: this variable corresponds to the first $\mathtt{T} \cdot \mathtt{U}$ variables $\{1, \dots, \mathtt{T}\} \times \{1, \dots, \mathtt{U}\}$ of λ_{TUF} in the MCMPP, i.e. it represents the flight profiles used in all the paths perform by UAV for going to tasks, but not the return path.
- *Return FP*: this variable corresponds to the last $\mathtt{U}$ variables of λ_{TUF} in the MCMPP, i.e. it represents the flight profiles used in the `ReturnToBase` paths (i.e. the $(\mathtt{T} + 1)$-th task) for every UAV used in the mission.

On the other hand, the auxiliary variables used in the CSP model are:

- *Departure*: This variable represents the time point for each UAV/task combination when the UAV departs from its position to go to the task zone.
- *Start*: This variable represents the time point for each UAV/task combination when the UAV arrives to the task zone and starts the performance of the task.
- *End*: This variable represents the time point for each UAV/task combination when the UAV finishes the task performance.
- *Return*: This variable represents the time point for each UAV when it has returned and landed into its base position, and consequently finished its duties in the mission.
- *Duration Path*, *Distance Path*, *Fuel Cons. Path* and *Path Waypoints*: These variables represent, respectively, the time elapsed, the distance traversed, the fuel consumed and the waypoints followed between the departure and the start of each UAV/task combination.
- *Duration Task*, *Distance Task*, *Fuel Cons. Task* and *Task Waypoints*: These variables represent, respectively, the time elapsed, the distance traversed, the fuel consumed and the waypoints followed between the start and the end of each UAV/task combination.
- *Duration Return*, *Distance Return*, *Fuel Cons. Return* and *Return Waypoints*: These variables represent, respectively, the time elapsed, the distance traversed, the fuel consumed and the waypoints followed between the end of the last task performed and the return for each UAV.
- *Duration Loiter*: This variable represents the time elapsed for each UAV/task combination between the end of the previous task performed by the UAV and the departure for the present task. This variable is higher than 0 when there exists some time dependencies that impede the immediate performance of the task.
- *UAV Paths*: This variable represents the waypoints followed by each UAV for the whole mission.
- *Fuel Consumptions*: This variable represents the fuel consumed by each UAV in the whole mission, i.e. $\mathrm{FUEL}_u(0) - \mathrm{FUEL}_u(T^{\max})$.
- *Flight Times*: This variable represents the flying time of each UAV during the whole mission, i.e. T_u.
- *Distances*: This variable represents the distance traversed by each UAV in the whole mission, i.e. D_u.
- *Costs*: This variable represents the cost spent for each UAV in the mission, i.e. $C_u T_u$.
- *Min Distance To Ground*: This variable represents the minimum altitude of each UAV to the ground (given by an elevation file) during its mission flight, i.e. $\min_{\tau \in [T_u^{Climb}, T_u^{Descent}]} A_u(\tau)$, where T_u^{Climb} and $T_u^{Descent}$ represent, respectively, the climb point when the UAV has reached the flight profile altitude after the take-off, and the descent point when the UAV starts the landing.

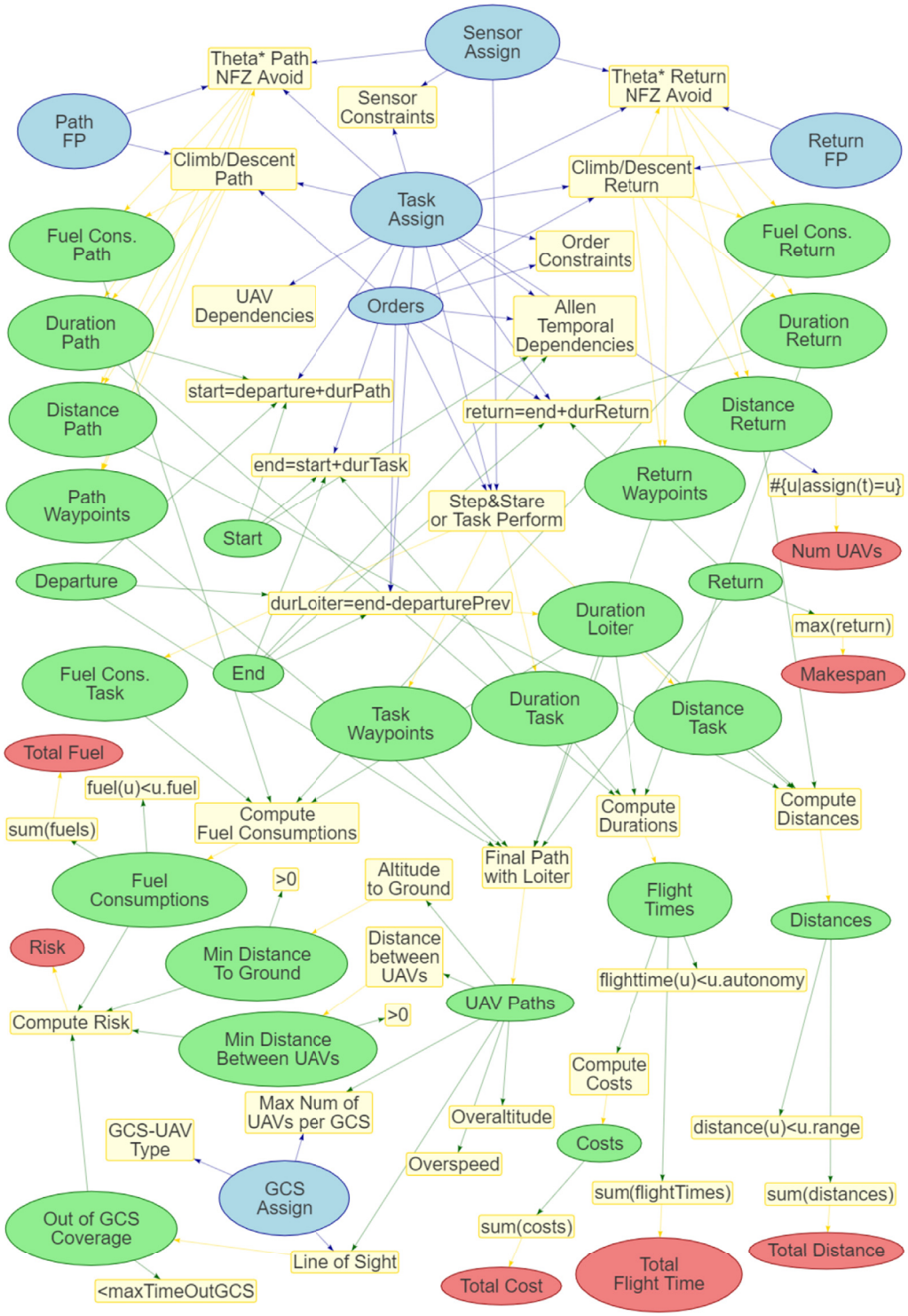

Fig. 2. Graph representation of the CSP model. Variables of the CSP are represented in blue, constraints in yellow, auxiliary variables in green and objectives in red. (For interpretation of the references to color in this figure legend, the reader is referred to the Web version of this article.)

- *Min Distance Between UAV*: This variable represents the minimum distance between each pair of UAV in the mission (lowest distance in all temporal points), i.e. $\min_{\tau\in[0,T^{\max}]}\|Pos_{u_1}(\tau)-Pos_{u_2}(\tau)\|_F$.
- *Out of GCS Coverage*: This variable represents the accumulated time for each UAV where LOS is not maintain with the GCS in the mission.

Finally, the constraints of the CSP considered are:

- *Sensor constraints*: they check if a UAV has the sensors needed to perform its assigned tasks. They are equivalent to Equation (3) in the MCMPP definition.
- *Order constraints*: they check that the orders for two different tasks assigned to the same UAV are different, and lower than the number of tasks assigned to that UAV, i.e. $\Gamma_{TU}(t_1,u)\neq\Gamma_{TU}(t_2,u)$, $\forall t_1\neq t_2\in\mathcal{T}\ \forall u\in\mathcal{U}$.

- *Allen Temporal Dependencies* and *UAV Dependencies*: these constraints are related to the time and vehicle dependencies mentioned previously and represented in Equation (12) in the MCMPP definition.
- *GCS constraints*: they assure that the GCS assignments are correct, UAV are assigned to GCS able to control them (*GCS-UAV Type*); GCS do not overpass their maximum number of UAV(*Max Num of UAV per GCS*) expressed in Equation (4); and they are located within the GCS coverage area (*Line of Sight*), expressed in Equation (5).
- *Path constraints*: these constraints assure that the UAV perform its path for every UAV/task combination without any incident. In this sense, a Theta* algorithm [24] is used to assure that the vehicle avoids the NFZ (*Theta* Path NFZ Avoid*), and that every climb/descent needed for the change of altitude (*Climb/Descent Path*) is attainable. These constraints are part of Equation (6) in the MCMPP definition.
- *Return constraints*: they assure that each UAV perform its return to the base without any incident, including NFZ avoidance (*Theta* Return NFZ Avoid*) and climb/descent routes (*Climb/Descent Return*). These constraints are part of Equation (6) in the MCMPP definition.
- *Step&Stare or Task Perform*: they compute the task performance waypoints, distances, durations and fuel consumptions according to the specific task, e.g. Mapping tasks perform a Step&Stare over the task zone.
- *Temporal constraints*: they assure the consistency of all the times and durations involved in the mission planning. This includes that $start = departure + durPath$, $end = start + durTask$, $return = end + durReturn$ and $durLoiter = departure - endPrev$.
- *Overaltitude* and *Overspeed*: these constraints assure that the waypoints for each UAV (*Final Path with Loiter*) do not overpass the speed and altitude limit of the vehicle.
- *Distance to ground* > 0: these constraints assure that the vehicle does not collide with the terrain. This is equivalent to Equation (7) in the MCMPP definition.
- *Distance between vehicles* > 0: they assure that UAV do not collide during the mission. This is equivalent to Equation (8) in the MCMPP definition.
- *Fuel constraints* ($fuel(u) < u.fuel$): they assure that the fuel consumed by each vehicle is less than its initial fuel. This is equivalent to Equation (9) in the MCMPP definition.
- *Flight time constraints* ($flighttime(u) < u.maxFlightTime$): they assure that the total flight time for each vehicle is less than its vehicle maximum flight time. This is equivalent to Equation (10) in the MCMPP definition.
- *Distance constraints* ($distance(u) < u.range$): they assure that the distance traversed by each vehicle is less than its range. This is equivalent to Equation (11) in the MCMPP definition.

3. Proposed weighted-random multi-objective evolutionary algorithm for mission planning

Real-world decision problems often require the solutions to meet multiple performance criteria (or objectives) simultaneously, for which they are referred to as MOP. Such objectives are often conflicting, wherein an improvement in one objective cannot be achieved without detriment to another objective. In this case, there is no single solution to a MOP that can be selected objectively; rather a set of solutions exists, representing different performance trade-offs between the considered criteria. A minimization MOP can be mathematically defined as follows:

$$\underset{\mathbf{x}}{\text{Min}}\ \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_m(\mathbf{x}))^T \quad \text{subject to } \mathbf{x} \in \Omega \subseteq \mathbb{R}^n \tag{13}$$

where $\mathbf{x} = (x_1, x_2, \dots, x_n)^T$ is a vector of n decision variables from the decision space Ω; $\mathbf{f} : \Omega \to \Theta \subseteq \mathbb{R}^m$ consists of a set of m objective functions, and a mapping from the n-dimensional decision space Ω to the m-dimensional objective space Θ.

Definition 1. Given two decision vectors $\mathbf{x}, \mathbf{y} \in \Omega$, $\mathbf{x}$ is said to *Pareto dominate* $\mathbf{y}$, denoted by $\mathbf{x} \prec \mathbf{y}$, if:

$$\forall i \in \{1, 2, \dots, m\} \quad f_i(\mathbf{x}) \le f_i(\mathbf{y})$$
$$\exists j \in \{1, 2, \dots, m\} \quad f_j(\mathbf{x}) < f_j(\mathbf{y}) \tag{14}$$

Definition 2. A decision vector $\mathbf{x}^* \in \Omega$, is *Pareto optimal* if $\nexists \mathbf{x} \in \Omega$, $\mathbf{x} \prec \mathbf{x}^*$.

Definition 3. The *Pareto set PS*, is defined as:

$$PS = \{\mathbf{x} \in \Omega \mid \mathbf{x} \text{ is Pareto optimal}\} \tag{15}$$

Definition 4. The *Pareto front PF*, is defined as:

$$PF = \{\mathbf{f}(\mathbf{x}) \in \mathbb{R}^m \mid \mathbf{x} \in PS\} \tag{16}$$

The goal of evolutionary multi-objective optimization (EMO) algorithms is to evolve the non-dominated objective vectors towards *PF* (convergence), and also generate a good distribution of these vectors over the *PF* (diversity). The most used algorithm over the last decade in this field has been the nondominated sorting genetic algorithm-III (NSGA-II) [26]. NSGA-II achieves convergence and diversity by relying on two measures when comparing individuals (e.g. for selection and deletion): The first is the non-domination rank, which measures how close an individual is to the non-dominated front. An individual with a lower rank (closer to the front) is always preferred to an individual with a higher rank. If two individuals have the same non-domination rank, as a secondary criterion, a crowding measure is used, which prefers individuals which are in rather deserted areas of the front. More precisely, for each individual the cuboid length is calculated, which is the sum of distances between an individual's two closest neighbours in each dimension. The individuals with greater cuboid length are then preferred. Fig. 3 shows an overview of these strategies.

In this approach, a previous hybrid approach based on MOEA and CSP [27] for the MCMPP is extended to consider the characteristics of the problem. The CSP is computed inside the fitness function of the MOEA, checking that solutions fulfill all the constraints. In addition, to deal with the huge search space of the problem and guide the algorithm to find valid solutions, a biased random generator has been designed for the initializer and mutation operator.

3.1. Encoding

The encoding of this new approach consists of six different alleles representing the features to be assigned in the MCMPP (presented in Section 2.1), as can be seen in Fig. 4:

1. *UAV assigned to each task.* This allele corresponds to the *Task Assign variable* of the CSP model. If the T_i task is Multi-UAV, then this cell contains a vector representing the different UAV assigned to this task, as shown in Fig. 4 for tasks T_1, T_3 and T_4. The number of combinations, or space size, for this allele is, considering T_{MU} the number of Multi-UAV tasks, $\mathrm{U}^{\mathrm{T}-\mathrm{T}_{MU}} \times (2^{\mathrm{U}} - 1)^{\mathrm{T}_{MU}}$.
2. *Flight profiles used for each UAV in the path to an assigned task.* This allele corresponds to the *Path FP variable* of the CSP model. As in the first allele, some of the cells could contain a vector if the corresponding task is performed by several UAV. The possible values considered in this approach are min (minimum consumption profile) and max (maximum speed profile). The space size for this allele is $2^{\mathrm{T}-\mathrm{T}_{MU}} \times (2^{\mathrm{U}+1} - 2)^{\mathrm{T}_{MU}}$.
3. *Sensors used for the task performance* by each UAV. This allele corresponds to the *Sensor Assign variable* of the CSP model. These variables are necessary when the vehicle performing the task has several sensors that could perform that task. The values considered in this approach are eiS (EO/IR sensor), sR (SAR radar), iR(ISAR radar) and mR (MPR radar). The space size for this allele depends on the num-

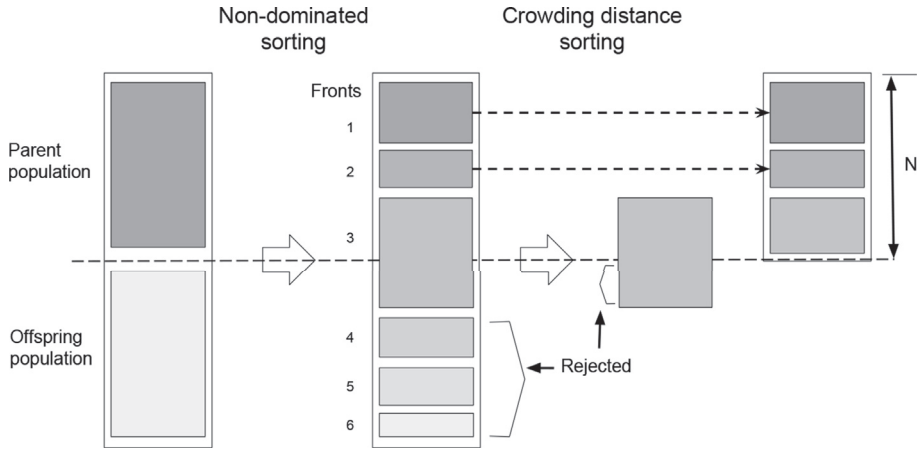

Fig. 3. NSGA-II overview of multiobjective strategies.

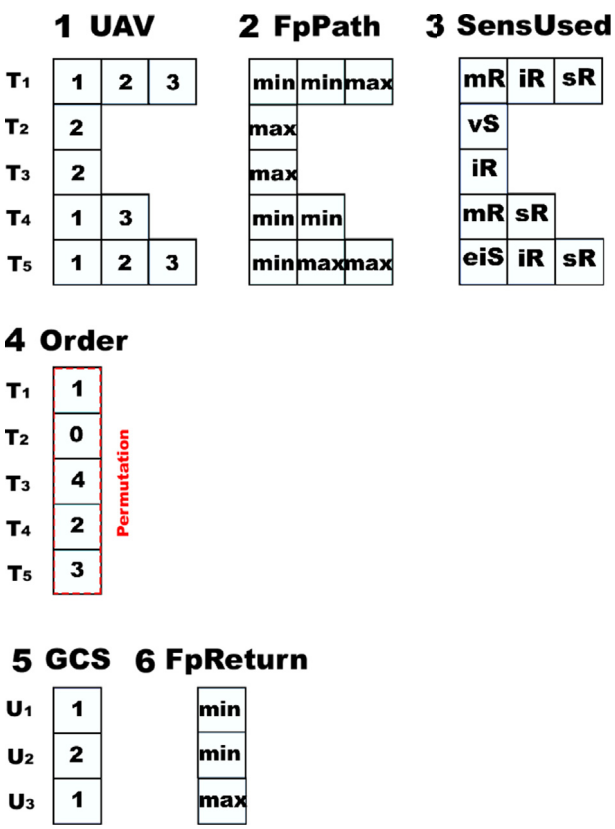

Fig. 4. Example of an individual that represents a possible solution for a problem with 5 tasks, 3 UAV and 2 GCS.

ber of sensors for each vehicle. Considering that all vehicles have s sensors, then the space size is $s^{\mathrm{T}-\mathrm{T}_{MU}} \times (\sum_{i=1}^{\mathrm{U}} s^i)^{\mathrm{T}_{MU}}$.

4. *Permutation of the task orders.* This allele corresponds to the *Orders variable* of the CSP model. These values indicate the absolute order of the tasks. It is only used if there are several tasks assigned to the same UAV (e.g. in Fig. 4, UAV 1 performs tasks 1, 4 and 5 in this order). In a previous publication [28], it was shown that this representation, although it is not injective, is more efficient than using the orders for each pair task-UAV. The space size for this allele is $\mathrm{T}!$.
5. *GCS controlling each UAV.* What GCS is assigned to monitor what UAV. This allele corresponds to the *GCS Assign variable* of the CSP model. The space size for this allele is G^{U}.
6. *Flight Profiles used by each* UAV to return to the base. This allele corresponds to the *Return FP variable* of the CSP model. The space size for this allele is 2^{U}.

The multiplication of the space sizes for each allele provides the total search space size of the problem, which shows the high complexity of adding a task, UAV or GCS to the problem.

In order to reduce this huge search space, some constraints have been used within the initialization and the mutation operator in order to assure that the solutions generated are more likely to be valid. These constraints considered are:

- *Sensor constraints*: In the task assignments, a UAV is selected if and only if it has the appropriate sensor to perform the task.
- *GCS-UAV type constraints*: In the GCS assignments, a GCS is selected for controlling a UAV if and only if the GCS is able to handle that type of vehicle.
- *Allen Temporal Dependencies*: In the order permutation, when generating the permutation, if there is some precedence or *meets* dependency, then this precedence is checked, and if not satisfied, these values are swapped in the permutation.

3.2. Weighted strategies for biased random generator

Usually, the first population for a MOEA is generated randomly. But sometimes, in order to reduce the convergence time, if some knowledge about the problem is provided, it can be used to guide the search of solutions.

This information can be expressed as a high probability but not absolute certainty of (e.g. in the vehicle routing problem (VRP) with multiple vehicles [29,30], it is highly probable that the best vehicle for going to a waypoint is the nearest to it in the space; but this is not always true as sometimes it implies other waypoints far from this but also near to the vehicle to remain unassigned due to fuel or flight time constraints). In order to consider this kind of information appropriately, weighted strategies can be used. These strategies are used instead of the typical random generator inside the MOEA, in both creation and mutation, and assigns different probabilities for each of the possible values of the genes.

In this paper, three strategies are proposed to be considered:

- Arithmetic strategy: This strategy gives a lower probability to less probable values following an inverse arithmetic progression (i.e. $\frac{N-i}{N}$, with N the biggest value and i the value considered). For an integer set of N values where the bigger the value, the lower its probability, the weight for each value using the arithmetic strategy will be as follows:

$$\forall i \in 1..N, \quad weight[i] = \frac{N-i}{N} \tag{17}$$

- Harmonic strategy: This strategy gives a lower probability to less probable values following a harmonic progression (i.e. $\frac{1}{i}$, with i the value considered). For an integer set of N values where the bigger the value, the lower its probability, the weight for each value using the harmonic strategy will be as follows:

$$\forall i \in 1..N, weight[i] = \frac{1}{i} \tag{18}$$

- Geometric strategy: This strategy gives a lower probability to less probable values following an inverse geometric progression (i.e. $\frac{1}{2^i}$, with i the value considered). For an integer set of N values where the bigger the value, the lower its probability, the weight for each value using the geometric strategy will be as follows:

$$\forall i \in 1..N, weight[i] = \frac{1}{2^i} \tag{19}$$

These different strategies are represented in Fig. 5 for a case with a maximum of 5 integer values with decreasing probability. In this figure, the constant strategy represents the typical random generator, where all values have the same probability.

Here, we can see that with the geometric approach, it is more likely to select value 1, while the harmonic and the arithmetic approach have lower probabilities. With the arithmetic approach, the decrease in the probabilities is linear, while in the geometric approach the decay is exponential. On the other hand, the harmonic approach provides a larger decrease for the first values, but lower for the last ones. Based on this, the use of the arithmetic approach is more suitable when the certainty of selecting the most likely value is not very high, so other less probable values could be easily selected. The harmonic approach should be instead used when the certainty of selecting the most probable value is high and selecting the less probable values is indifferent for the final result. Finally, the geometric approach becomes useful when the certainty of selecting the most probable value is very high, and the less certain any other value is, the less likely is to obtain good results using this weight distribution.

In this approach, we have implemented a biased random generator for the formation of new individuals. This biased random generator is applied in three parts of the encoding:

- *Number of UAV Selection (NUS)*: for Multi-UAV tasks, the selection of the number of UAV is performed using a weighted strategy. In the previous hybrid approach, most of the optimization objectives (except for the makespan) used as fitness function, obtained better results when the number of UAV assigned for these tasks is low. For this reason, the new weighted random function provides a lower probability for higher numbers of UAV.
- *Distance-based UAV Selection (DUS)*: for any task, the selection of the UAV(s) to perform it, is done through a weighted strategy dependant on the distance from the UAV(s) to the task. In each mission, there are different vehicles that can perform the same task. Due to this, the selection of the UAV(s) commanded to perform a specific task could hinge on the distance of every UAV(s) to the task at hand. UAV(s) closer to the task would spent less time to arrive to the location where the task must be performed, and therefore obtain better solutions with less fuel consumption, flight time, etc. So, the closer the UAV is to the task, the higher the probability of being selected.
- *Distance-based GCS Selection (DGS)*: for any UAV, the selection of the GCS controlling it, is done through a weighted strategy dependant on the distance from the UAV to the GCS. The closer the GCS is to the UAV, the higher the probability of being selected.

In order to assume different orders of consistency for the selection of the UAV number, several strategies have been proposed for the NUS weighted random function:

- Arithmetic: $\forall i = 1..\mathtt{U}, weight[i] = \frac{\mathtt{U}-i}{\mathtt{U}}$
- Harmonic: $\forall i = 1..\mathtt{U}, weight[i] = \frac{1}{i}$
- Geometric: $\forall i = 1..\mathtt{U}, weight[i] = \frac{1}{2^i}$

These different strategies are represented in Fig. 5 for a case with a maximum of 5 UAV. Here, we can see that with the geometric approach, it is more likely to select just 1 UAV, while the harmonic and the arithmetic approach have lower probabilities. The constant approach (uniform random function) gives the same probability to every possible number of UAV to be used.

On the other hand, the same strategies have been implemented for the DUS weighted random function. However, in this case instead of the number of UAV, the distance from each one to the task is used:

- Arithmetic: $\forall t \in \mathcal{T}, \forall u \in \mathcal{U}, weight[t,u] = \frac{\max_{j\in\mathcal{U}}\|Pos_u(0)-Pos_j\|_F - \|Pos_u(0)-Pos_t\|_F}{\max_{j\in\mathcal{U}}\|Pos_u(0)-Pos_j\|_F}$
- Harmonic: $\forall t \in \mathcal{T}, \forall u \in \mathcal{U}, weight[t,u] = \frac{1}{\|Pos_u(0)-Pos_t\|_F}$
- Geometric: $\forall t \in \mathcal{T}, \forall u \in \mathcal{U}, weight[t,u] = \frac{1}{2^{\|Pos_u(0)-Pos_t\|_F}}$

Finally, in a similar way, these strategies are used with the DGS weighted random function:

- Arithmetic: $\forall u \in \mathcal{U}, \forall g \in \mathcal{G}, weight[u,g] = \frac{\max_{j\in\mathcal{G}}\|Pos_u(0)-Pos_j\|_F - \|Pos_u(0)-Pos_g\|_F}{\max_{j\in\mathcal{G}}\|Pos_u(0)-Pos_j\|_F}$
- Harmonic: $\forall u \in \mathcal{U}, \forall g \in \mathcal{G}, weight[u,g] = \frac{1}{\|Pos_u(0)-Pos_g\|_F}$
- Geometric: $\forall u \in \mathcal{U}, \forall g \in \mathcal{G}, weight[u,g] = \frac{1}{2^{\|Pos_u(0)-Pos_g\|_F}}$

3.3. Algorithm

The weighted-random NSGA-II (WRNSGA-II) is presented in Fig. 6. In this new approach, firstly a *weighted random generation of the initial population* is performed. Then this initial population evolves using a multi-objective evolutionary algorithm based on NSGA-II approach [26].

The population weighted random initialization generates a set of λ weighted random individuals, each one generated using the function presented in Algorithm 1. First, the number of UAV to be used when tasks are Multi-UAV is selected (Lines 4–8) using the provided NUS strategy. Then, the concrete UAV to be used are selected (Lines 10–14)

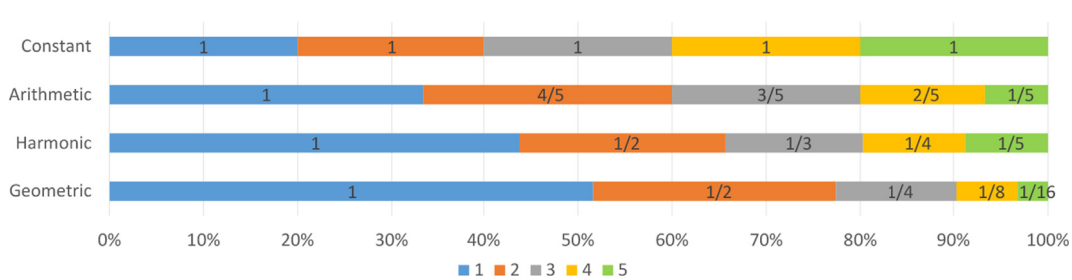

Fig. 5. Weighted strategies considered with 5 integer values of decreasing probability.

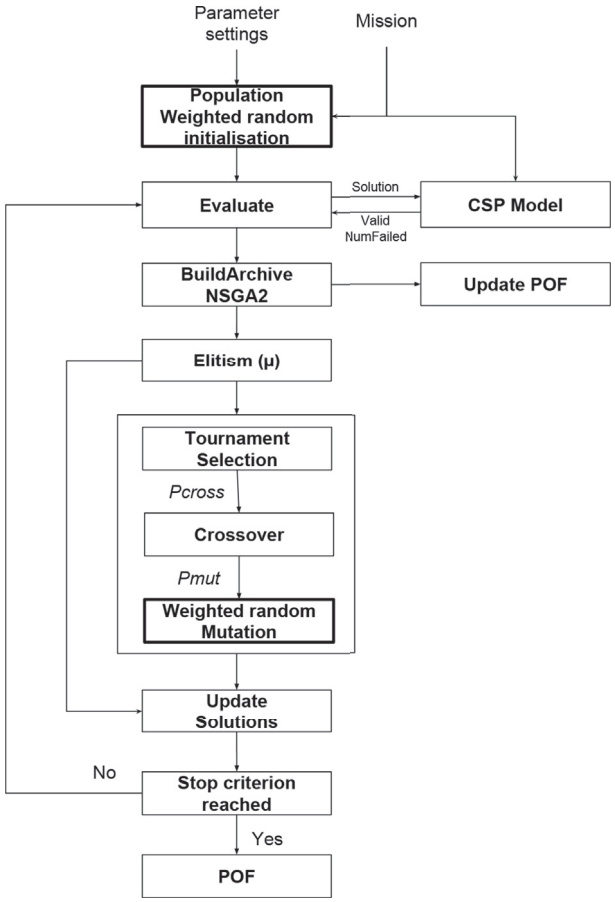

Fig. 6. Overview of WRNSGA-II algorithm for mission planning.

according to the provided DUS strategy. After that, the permutation orders, path flight profiles and sensors used are selected randomly as usual. In Lines 20–23, it is shown how the GCS controlling the vehicles are selected according to the DGS strategy. In this algorithm, the costs computed for strategies are represented as *cNUS*, *cDUS* and *cDGS*. In both DUS and DGS, the geodesic distance function used to compute the distances between tasks, vehicles and ground stations is represented as *d*.

For all these selections, a general strategy function returning the cost for each value has been designed (Algorithm 2), as well as a weighted random function which returns a weighted random value according to the costs provided (Algorithm 3).

The evaluation of the individuals is performed by a fitness function that first checks that all the constraints of the MCMPP are fulfilled for a given solution. For this, a CSP solver is used to check if the constraints are satisfied for a concrete solution. When the solution is not valid according to the CSP, then the number of unfulfilled constraints is returned, and used as the optimization criteria with other invalid solutions. On the other hand, if all constraints are fulfilled, the fitness works as a multi-objective function minimizing the objectives of the problem presented in Section 2.

A tournament selection is used to provide the individuals that will be chosen to apply the genetic operators. The crossover operator consists of an extension of the 2-point crossover and the partially-matched crossover (PMX) independently applied to each allele of the chromosome, where a 2-point crossover is applied to the 3 first alleles of size $\mathtt{T} \cdot \mathtt{U}$, PMX is applied to the fourth allele (the permutation of orders), and another 2-point crossover is applied to the fifth and sixth alleles of size $\mathtt{U}$. The mutation operator is also an extension of the uniform and the insert mutation applied to each allele using the same criterion as in the crossover. This *mutation operator* is *combined* with the *biased random generator* in order to produce new individuals based on the weighted random functions previously explained.

Finally, the stopping criteria designed for this algorithm compares the non dominated solutions obtained so far at each generation with the solutions from the previous one. If the solutions from a previous generation remains unchangeable for a number of generations, then the algorithm will stop and return these solutions as an approximation of the POF.

4. Experimental results

In these experiments, the main goal is to test which of the strategies proposed (and previously defined in Section 3.3) best fits for each weighted random functions (NUS, DUS and DGS) used in a particular set of mission planning problems. The experiments are divided in two parts: first, the different NUS, DUS and DGS strategies are applied and compared independently, i.e. the different NUS strategies are compared between them to get the best one, as well as the different DUS and the different DGS strategies. These strategies are compared against the previous approach, which is represented as the *constant strategy*. Then, the best of these strategies for each part are selected and joined to obtain the best combination of strategies. This combination is tested and compared against the previous results.

For this purpose, a set of 16 missions of different complexity has been designed. These datasets are presented in Table 1, where the different number of tasks (and Multi-UAV tasks), UAV, GCS, NFZ and time dependencies are shown.

The setup parameters of the WRNSGA-II are as follows. The selection criteria ($\mu + \lambda$) used was 30 + 300, where λ is the offspring size, and μ the elitism size, i.e. the number of the best parents that survive from the current generation to the next one. The mutation probability is 10%, and the number of generations used in the stopping criteria is 10. Each problem is run 10 times, and the average and standard deviation of the hypervolume [31] of the obtained solutions are calculated. The hypervolume has been computed using the normalized values of the objectives variables. Moreover, the number of generations needed to converge are extracted for each case. Instead of studying the computation time directly, which depends on the hardware and programming language used, the number of generations needed to converge, (which are proportional to the computation time) are studied in these experiments.

For the sake of validating the statistical significance of the obtained results, the distributions of the metric values obtained by the different methods on each dataset have been compared by means of a nonpara-

Table 1
Features of the different datasets designed.

Dataset Id.	Tasks	Multi-UAV Tasks	UAV	GCSs	NFZs	Time Dependencies
1	5	0	3	1	0	0
2	6	1	3	1	1	0
3	6	1	4	2	2	1
4	7	1	5	2	1	2
5	8	2	5	2	3	1
6	9	2	5	2	0	2
7	9	2	6	2	2	2
8	10	2	6	2	3	3
9	11	3	6	2	3	2
10	12	3	7	3	0	2
11	12	3	8	3	2	3
12	13	4	7	3	4	4
13	14	4	8	3	0	3
14	15	4	9	3	5	4
15	16	4	9	3	4	4
16	16	5	10	3	5	5

Algorithm 1 WeightedRandomIndividual (*M*, *NUS*, *DUS*, *DGS*).

Input: A mission $M = (\mathcal{T}, \mathcal{U}, \mathcal{G})$ where $\mathcal{T}$ is a set of tasks to perform denoted by $\{t_1, \ldots, t_{\mathtt{T}}\}$, $\mathcal{U}$ is a set of UAV denoted by $\{u_1, \ldots, u_{\mathtt{U}}\}$ and $\mathcal{G}$ is a set of GCSs denoted by $\{g_1, \ldots, g_{\mathtt{G}}\}$. And weighted strategies for NUS, DUS and DGS

Output: A Weighted random individual

```
ind ← ∅
for t_i ∈ T do
    num_UAV ← 1
    if isMultiUAV(t_i) then
        cNUS ← [Strategy(j, U, NUS)]_{j=1}^{U}
        num_UAV ← weightedRandomValue([1,2,...,U], cNUS)
    t_assign ← ∅
    while |t_assign| < num_UAV do
        cDUS ← [Strategy(d(t_i, u_j), max_{u_k∈U} d(t_i, u_k), DUS)]_{u_j∈U}
        t_assign ← t_assign ∪ weightedRandomValue(U, cDUS)
    ind.UAV[t_i] ← t_assign
assignPermutationOrders(ind, M)
assignFPPaths(ind, M)
assignSensors(ind, M)
for u_i ∈ U do
    cDGS ← [Strategy(d(u_i, g_j), max_{g_k∈G} d(u_i, g_k), DGS)]_{g_j∈G}
    ind.GCS[u_i] ← weightedRandomValue(G, cDGS)
assignFPReturns(ind, M)
return ind
```

Algorithm 2 Strategy (*value*, *max*, *strategy*).

Input: A *value* to which the concrete *strategy* must be applied, and the maximum value *max* between all possible values.

Output: The concrete cost for the value

```
switch strategy do
    case Arithmetic do
        return (max − value) / max
    case Harmonic do
        return 1 / value
    case Geometric do
        return 1 / 2^value
    case Constant do
        return 1
```

Algorithm 3 WeightedRandomValue (values, costs).

Input: Vectors V (integer values considered) and C (double costs for each value) of size d

Output: A weighted random value based on the costs

```
total ← sum(C)
randomValue ← randomDouble() · total
for i ← 1 to d do
    randomValue ← randomValue − C_i
    if randomValue ≤ 0 then
        return V_i
```

Table 2
Comparative assessment of the different weighted strategies used in NUS. Results show the average ± standard deviation for the hypervolume and the number of generations. For each dataset the best obtained result is marked in bold.

Id.	Constant Strategy		Arithmetic Strategy		Harmonic Strategy		Geometric Strategy	
	Hypervol.	No. Gen.	Hypervol.	No. Gen.	Hypervol.	No. Gen.	Hypervol.	No. Gen.
1	**$0.80 \pm 1e^{-6}$**	41 ± 6	$0.80 \pm 1e^{-6}$	46 ± 7	$0.80 \pm 1e^{-6}$	**41 ± 5**	$0.80 \pm 1e^{-6}$	44 ± 7
2	$0.72 \pm 1e^{-3}$	253 ± 19	$0.72 \pm 2e^{-3}$	243 ± 25	**$0.72 \pm 1e^{-3}$**	**231 ± 23**	$0.72 \pm 3e^{-3}$	233 ± 17
3	0.69 ± 0.01	395 ± 26	0.68 ± 0.03	382 ± 27	**0.69 ± 0.01**	368 ± 30	0.69 ± 0.02	**363 ± 25**
4	0.74 ± 0.02	593 ± 46	0.74 ± 0.03	578 ± 35	0.73 ± 0.04	543 ± 37	**0.74 ± 0.01**	**537 ± 21**
5	0.68 ± 0.04	408 ± 32	0.67 ± 0.06	393 ± 36	0.68 ± 0.03	376 ± 29	**0.69 ± 0.02**	**370 ± 31**
6	0.63 ± 0.04	697 ± 38	0.63 ± 0.04	654 ± 39	0.63 ± 0.05	626 ± 36	**0.63 ± 0.03**	**601 ± 38**
7	0.57 ± 0.08	865 ± 41	**0.57 ± 0.07**	812 ± 39	0.57 ± 0.08	777 ± 36	0.57 ± 0.07	**759 ± 37**
8	0.40 ± 0.10	968 ± 31	0.42 ± 0.08	928 ± 35	0.43 ± 0.10	880 ± 33	**0.43 ± 0.08**	**846 ± 26**
9	0.26 ± 0.12	1000 ± 1	0.31 ± 0.14	998 ± 3	0.33 ± 0.13	976 ± 15	**0.33 ± 0.11**	**951 ± 18**
10	0.15 ± 0.08	1000 ± 0	0.23 ± 0.14	1000 ± 1	0.26 ± 0.10	997 ± 2	**0.28 ± 0.09**	**992 ± 4**
11	0.05 ± 0.06	1000 ± 0	0.12 ± 0.10	1000 ± 0	0.19 ± 0.09	1000 ± 0	**0.24 ± 0.04**	1000 ± 0
12	$1e^{-3} \pm 2e^{-3}$	1000 ± 0	0.03 ± 0.01	1000 ± 0	0.05 ± 0.01	1000 ± 0	**0.11 ± 0.06**	1000 ± 0
13	$2e^{-5} \pm 3e^{-5}$	1000 ± 0	$3e^{-3} \pm 3e^{-4}$	1000 ± 0	$0.01 \pm 6e^{-3}$	1000 ± 0	**0.06 ± 0.01**	1000 ± 0
14	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0	$2e^{-3} \pm 7e^{-4}$	1000 ± 0	**$0.01 \pm 2e^{-3}$**	1000 ± 0
15	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0
16	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0

metric Kruskal-Wallis test [32]. This test represents the nonparametric version of the classical one-way ANOVA and is an extension of the Wilcoxon rank sum test to groups larger than 2. To this end, the test compares the medians of the group, and returns the P value for the null hypothesis that all samples are drawn from the same population (or equivalently, from different populations with the same distribution). If the P value is lower than a predefined α, we can infer that the null hypothesis does not hold, that is, at least one sample median in the group is significantly different from the others, with $(1 - \alpha)$ level of confidence, then to determine which sample medians are statistically different, we have applied this multiple comparison procedure with $\alpha = 0.05$ (thus, with a 95% level of confidence).

In the first experiment, a comparative assessment of the different strategies for the NUS weighted random function has been carried out, as shown in Table 2. These results show that the new strategies for the NUS weighted random function help reducing the number of generations needed to converge while maintaining the hypervolume of the solutions. Concretely, the effects of these strategies are specially appreciated in datasets with a bigger number of Multi-UAV tasks. The statistical significance of the results could be checked through the Kruskal-Wallis test, showing a significant relevance ($p < 0.05$) for the results obtained in terms of number of generations needed to converge for datasets 4–9, while a significant relevance is provided for datasets 10–14 in terms of hypervolume. In datasets 1, the results are quite the same for the different strategies, so they are irrelevant, as datasets 2 and 3, which did not reach a significance relevance through the Kruskall-Wallis test. Datasets 15 and 16, on the other hand, did not reach any solution. It can be also appreciated as dataset 14, which did not find any solution using the common NSGA-II (constant strategy), could find solutions in some of the executions with the harmonic and geometric strategies. In general, the geometric strategy got better results in terms of number of generations needed to converge (it was the fastest approach), so we selected this strategy for the NUS weighted random function.

We next proceed by analyzing the results of the experiments performed with the different strategies for the DUS weighted random function, which are presented in Table 3.

As can be seen, these results show that applying any of the strategies highly outperformed the convergence of the algorithm due to the guidance of the search. The Kruskal-Wallis test confirmed the significance relevance of the different strategies compared to the common NSGA-II (constant strategy) in either terms of number of generations and/or hypervolume, except for dataset 16, which did not converge in any case. In the first datasets it can be appreciated that the geometric strategy attains the best results, but as the complexity of the problem increases, the hypervolume for this strategy gets lower, probably due to

Table 3
Comparative assessment of the different weighted strategies used in DUS. Results show the average ± standard deviation for the hypervolume and the number of generations. For each dataset the best obtained result is marked in bold.

Id.	Constant Strategy		Arithmetic Strategy		Harmonic Strategy		Geometric Strategy	
	Hypervol.	No. Gen.	Hypervol.	No. Gen.	Hypervol.	No. Gen.	Hypervol.	No. Gen.
1	$0.80 \pm 1e^{-6}$	41 ± 6	$0.80 \pm 2e^{-6}$	40 ± 6	**$0.80 \pm 1e^{-6}$**	**39 ± 5**	$0.80 \pm 2e^{-6}$	39 ± 6
2	$0.72 \pm 1e^{-3}$	253 ± 19	$0.72 \pm 2e^{-3}$	249 ± 25	**$0.72 \pm 1e^{-3}$**	240 ± 23	$0.72 \pm 3e^{-3}$	**238 ± 17**
3	0.69 ± 0.01	395 ± 26	0.69 ± 0.02	366 ± 24	**0.69 ± 0.01**	354 ± 18	0.68 ± 0.03	**345 ± 17**
4	0.74 ± 0.02	593 ± 46	0.73 ± 0.04	566 ± 29	**0.74 ± 0.02**	531 ± 33	0.73 ± 0.02	**519 ± 25**
5	**0.68 ± 0.04**	408 ± 32	0.68 ± 0.05	365 ± 30	0.67 ± 0.03	351 ± 26	0.64 ± 0.02	**342 ± 25**
6	0.63 ± 0.04	697 ± 38	0.62 ± 0.03	621 ± 29	**0.63 ± 0.04**	572 ± 41	0.61 ± 0.03	**539 ± 63**
7	0.57 ± 0.08	865 ± 41	**0.57 ± 0.07**	789 ± 40	0.56 ± 0.08	751 ± 34	0.51 ± 0.11	**710 ± 72**
8	0.40 ± 0.10	968 ± 31	0.43 ± 0.08	879 ± 37	**0.44 ± 0.10**	848 ± 32	0.40 ± 0.13	**825 ± 69**
9	0.26 ± 0.12	1000 ± 1	0.40 ± 0.10	975 ± 15	**0.42 ± 0.09**	929 ± 45	0.38 ± 0.15	**903 ± 89**
10	0.15 ± 0.08	1000 ± 0	0.29 ± 0.09	996 ± 3	**0.33 ± 0.10**	972 ± 12	0.29 ± 0.15	**921 ± 78**
11	0.05 ± 0.06	1000 ± 0	0.19 ± 0.10	1000 ± 0	**0.25 ± 0.08**	1000 ± 0	0.21 ± 0.13	**956 ± 98**
12	$1e^{-3} \pm 2e^{-3}$	1000 ± 0	0.12 ± 0.12	1000 ± 0	**0.19 ± 0.07**	986 ± 69	0.15 ± 0.13	**936 ± 123**
13	$0 \pm 3e^{-5}$	1000 ± 0	0.04 ± 0.01	1000 ± 0	**0.10 ± 0.02**	972 ± 54	0.06 ± 0.01	**902 ± 92**
14	0 ± 0	1000 ± 0	$5e^{-3} \pm 4e^{-3}$	997 ± 25	0.02 ± 0.01	982 ± 68	**$0.03 \pm 4e^{-3}$**	**897 ± 142**
15	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0	$1e^{-3} \pm 4e^{-3}$	967 ± 101	**$0.01 \pm 2e^{-3}$**	**902 ± 168**
16	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0

Table 4
Comparative assessment of the different weighted strategies used in DGS. Results show the average ± standard deviation for the hypervolume and the number of generations. For each dataset the best obtained result is marked in bold.

Id.	Constant Strategy		Arithmetic Strategy		Harmonic Strategy		Geometric Strategy	
	Hypervol.	No. Gen.	Hypervol.	No. Gen.	Hypervol.	No. Gen.	Hypervol.	No. Gen.
1	0.80 ± $1e^{-6}$	**41 ± 6**	0.80 ± $2e^{-6}$	44 ± 5	0.80 ± $1e^{-6}$	42 ± 6	**0.80 ± $1e^{-6}$**	45 ± 5
2	0.72 ± $1e^{-3}$	253 ± 19	**0.72 ± $1e^{-3}$**	256 ± 15	0.72 ± $3e^{-3}$	**251 ± 18**	0.71 ± $2e^{-3}$	255 ± 20
3	**0.69 ± 0.01**	395 ± 26	0.68 ± 0.01	390 ± 20	0.69 ± 0.02	388 ± 17	0.69 ± 0.02	**383 ± 18**
4	0.74 ± 0.02	593 ± 46	**0.74 ± 0.01**	591 ± 40	0.74 ± 0.02	573 ± 38	0.74 ± 0.01	**560 ± 31**
5	0.68 ± 0.04	408 ± 32	**0.68 ± 0.03**	398 ± 34	0.68 ± 0.04	385 ± 36	0.67 ± 0.04	**379 ± 27**
6	0.63 ± 0.04	697 ± 38	**0.63 ± 0.03**	676 ± 38	0.63 ± 0.04	641 ± 36	0.62 ± 0.02	**620 ± 43**
7	0.57 ± 0.08	865 ± 41	0.57 ± 0.06	812 ± 35	**0.57 ± 0.06**	795 ± 40	0.55 ± 0.07	**778 ± 55**
8	0.40 ± 0.10	968 ± 31	0.41 ± 0.10	938 ± 36	**0.43 ± 0.09**	901 ± 40	0.42 ± 0.10	**889 ± 47**
9	0.26 ± 0.12	1000 ± 1	0.30 ± 0.09	983 ± 9	**0.32 ± 0.08**	962 ± 15	0.30 ± 0.10	**946 ± 30**
10	0.15 ± 0.08	1000 ± 0	0.26 ± 0.09	998 ± 2	**0.29 ± 0.08**	986 ± 9	0.26 ± 0.11	**942 ± 56**
11	0.05 ± 0.06	1000 ± 0	0.17 ± 0.10	1000 ± 0	**0.21 ± 0.09**	1000 ± 0	0.19 ± 0.11	**976 ± 89**
12	$1e^{-3} \pm 2e^{-3}$	1000 ± 0	0.09 ± 0.06	1000 ± 0	**0.13 ± 0.08**	1000 ± 0	0.10 ± 0.10	**967 ± 102**
13	0 ± $3e^{-5}$	1000 ± 0	0.02 ± $8e^{-3}$	1000 ± 0	**0.07 ± 0.01**	1000 ± 0	0.03 ± $5e^{-3}$	**938 ± 103**
14	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0
15	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0
16	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0	0 ± 0	1000 ± 0

local minima. For this purpose the harmonic strategy gets better results, so it will be regarded as the best-performing strategy for the final experiment.

Then, we consider the DGS weighted random function. The results of the experiments performed with the different strategies for this function are presented in Table 4.

In this case, the behaviour is similar to DUS strategies, where the geometric strategy is better for the first datasets, but for more complex datasets, it gets stuck in local minima. The Kruskal-Wallis test is once more applied to confirm the significance relevance of the different strategies compared to the common NSGA-II in either terms of number of generations and/or hypervolume, showing relevance in datasets 5–13. So, as before, the harmonic strategy gets better results for more complex datasets, so it will be taken as the better strategy for the final experiment.

In the final experiment, we took the best strategies obtained in the previous experiments (geometric for NUS, harmonic for DUS and harmonic for DGS) and used the all together in the algorithm. The results obtained are shown in Table 5, where the average, standard deviation and p-value obtained in the Kruskall-Wallis test (in comparison with the common NSGA-II approach with no strategies applied) are presented for both the hypervolume obtained and the number of generations needed to converge.

Table 5
Results (average/standard deviation/Kruskall-Wallis p-value) obtained with the best combination of strategies (geometric NUS, harmonic DUS and harmonic DGS).

Id.	Hypervolume	No. Generations
1	0.80/$1e^{-6}$/0.76	39/7/0.31
2	0.71/$4e^{-3}$/0.56	203/12/0.06
3	0.69/0.03/0.65	274/21/0.03
4	0.74/0.02/0.54	486/24/0.04
5	0.68/0.05/0.48	334/29/0.03
6	0.63/0.02/0.45	519/37/0.03
7	0.56/0.10/0.54	703/41/0.04
8	0.48/0.09/0.41	808/34/0.04
9	0.51/0.08/0.36	876/31/0.03
10	0.36/0.09/0.09	916/26/0.04
11	0.28/0.12/0.05	987/9/0.02
12	0.31/0.07/0.04	998/3/$4e^{-3}$
13	0.26/0.03/0.04	1000/1/$7e^{-5}$
14	0.29/0.01/0.04	1000/0/1.0
15	0.16/$5e^{-3}$/0.04	1000/0/1.0
16	0.06/$5e^{-3}$/0.04	1000/0/1.0

These results show a notable improvement in the reduction of the number of generations needed to converge for complex datasets, which reflect an increased convergence rate. In addition, some valid solutions were obtained for dataset 16, where other approaches did not reach any solution. As the stopping criteria used for convergence checks that the non-dominated solutions obtained remain unchangeable for 10 generations, it is common that for the most complex datasets, with a lot of solutions in the POF, the algorithm did not reach the complete POF before the 1000 generations limit.

As there are 7 objectives to optimize, in order to visualize the form of the Pareto fronts obtained for each dataset, a parallel coordinates visualization [33] has been used to represent the solutions obtained. In order to ease the visualization and interpretation of the interplay between the different objectives, z-scores are used in these parallel plots, so changes are smoother depending on the standard deviation of the values for each objective.

For instance, in Fig. 7, the parallel coordinates visualization for dataset 4 is presented, where we can clearly distinguish several sets of solutions that have been clustered in different colors. In blue, it is appreciable that there are some solutions that have a high makespan, but instead have low values for the rest of variables. The red solutions are similar, but decreasing the makespan value while increasing the others except for the risk, fuel and number of UAV, which remain the lowest. Green solutions have the highest values for cost, flight time and distance, while preserving a medium value for makespan, fuel, risk and number of UAV. Purple solutions present the highest values for fuel and risk, but have the lowest values for makespan. Orange solutions also have low values for makespan and high values for risk and number of UAV, while the rest of variables have medium-high values. Finally, brown and yellow solutions have medium values for most variables, except for risk in some cases.

As the complexity of the problem increases, there are more non-dominated solutions and most times they are more scattered, so the visualization becomes harder. For example, Fig. 8 shows the parallel coordinates visualization for dataset 9. Here, it is more difficult to appreciate sets of solutions. In blue, some solutions with low cost, fuel and distance, but high risk and medium number of UAV are represented. Green solutions present a high value for fuel, while maintaining low values for flight time and number of UAV. Yellow solutions present a low number of UAV but a high risk. Purple solutions have high cost, flight time and distance, but low risk. Red solutions have the highest makespan, while orange solutions have the lowest, but the rest of variables are very scattered for these solutions.

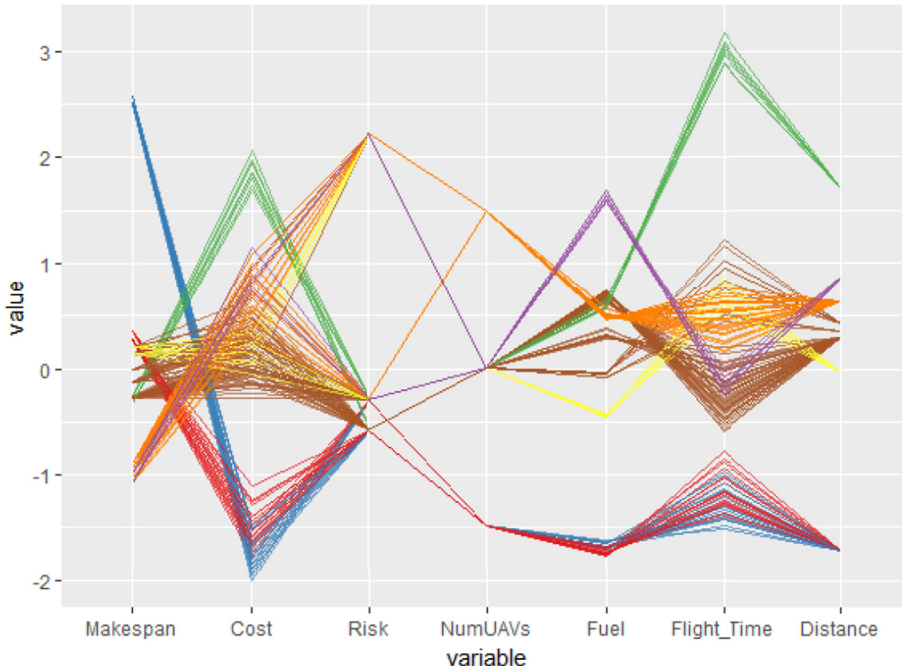

Fig. 7. Parallel Coordinates Visualization of the solutions obtained for dataset 4.

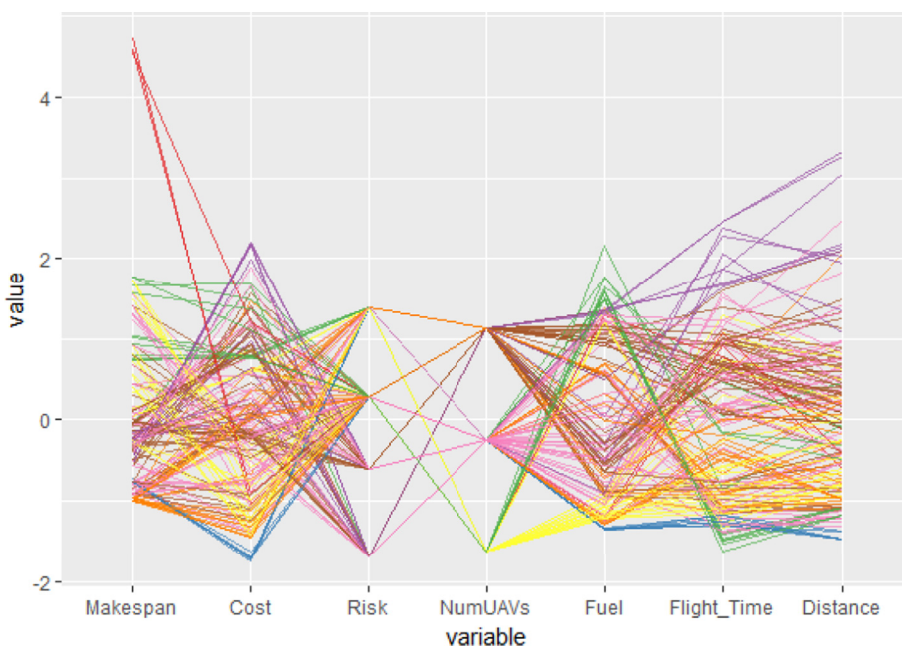

Fig. 8. Parallel coordinates visualization of the solutions obtained for dataset 9.

In these cases, sometimes is useful to use another type of visualization, such as radial visualization (RadViz) [34], in order to visualize the clusterization of the solutions. The RadViz plot for dataset 9 can be seen in Fig. 9, maintaining the same colors used in Fig. 8. In this plot, we can see that the same solutions clustered in the parallel coordinates approach are grouped together in RadViz (except for pink solutions, which represent the remaining solutions that were not clustered). In this visualization, we can appreciate how some sets of solutions approach to some objectives (when they have higher values) while they keep away from others (when they have lower values), e.g. blue solutions are very close to risk and number of UAV, because these solutions are worse for these variables, while they stay far from the rest of variables, were they performed better. In this sense, the solutions more centered in the graph, represent the ones having middle values for most variables, e.g. some pink solutions and the purple solutions. On the other hand, the yellow solutions keep away from the number of UAV, which are better optimized, but we can appreciate how some of them stay close to the cost, while other approach to the makespan, because they get different results in these two variables. Besides, the red solutions clearly approach the makespan, were they have the biggest value for this variable. It can be noticed that some color/solution pair sets are merged together, such as yellow and green solutions, or orange and brown solutions. This is due to the big similarities between these groups, were only one or two variable behave differently, and usually these variables still

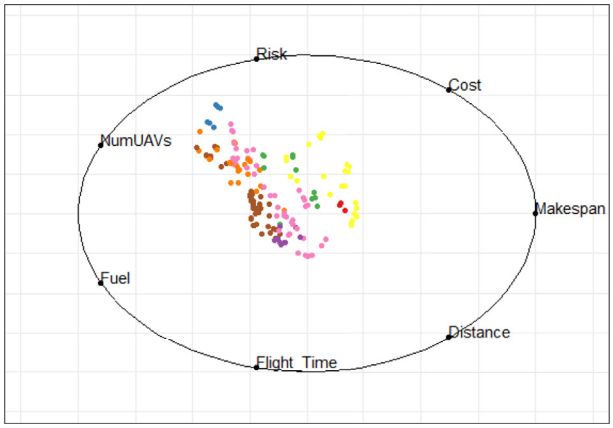

Fig. 9. Radial visualization (RadViz) of the solutions obtained for dataset 9.

remain close, with no big numeric difference between them, e.g. green and yellow are mostly similar except for some yellow solutions that get better results in terms of fuel and cost, while green solutions do not.

With the combination of these two visualization methods, we can conclude that for complex datasets, as dataset 9, where there is a big number of solutions and they are highly sparsed, it is possible to appreciate some behaviors in terms of clustering of solutions, although most of these clusters are merged between them.

5. Conclusions

In this paper we have presented a new weighted random MOEA-CSP approach for solving the Multi-UAV mission planning problem. The presented approach considers missions consisting of several tasks to be performed by several UAV using a specific sensor. Each UAV is controlled by a GCS and use specific flight profiles. The problem has been modelled as a CSP where the encoding of the MOEA considers the same variables as the CSP. Thereby, the fitness function can use the CSP to check if solutions are valid, and then a multi-objective fitness function optimizing several objectives is considered.

In addition, a biased random generator has been designed in order to guide the generation of new individuals. This generator improves the selection of the number of UAV for Multi-UAV tasks, and the selection of the UAV based on the distance to the task. For each selection, three strategies (arithmetic, harmonic and geometric) have been proposed to be used as weighted random functions.

The experiments performed over several missions show that this new approach outperforms the results obtained previously in terms of increased convergence rate. In addition, when the scenario of the mission is not favourable for the proposed weighted random approach, we obtain similar results to the previous approach. Moreover, this approach has resulted very helpful for problems of big complexity where convergence is difficult.

The main lack in this approach is the huge number of solutions obtained when missions are complex, which makes it difficult to select the final solution to be executed by the operator. Future works will focus on developing a decision 838 support system (DSS) for this problem, in order to select a few solutions among those obtained by the algorithm according to some quality metrics and the GCS operator profile.

Acknowledgements

This work has been co-funded by: Savier Project (Airbus Defence & Space, FUAM-076915), Spanish Ministry of Science and Education and Competitivity (MINECO) and European Regional Development Fund (FEDER) under projects EphemeCH (TIN2014-56494-C4-4-P), and DeepBio (TIN2017-85727-C4-3-P), Comunidad Autónoma de Madrid under project CIBERDINE S2013/ICE-3095, and the Basque Government for its support through the EMAITEK program. The authors would like to acknowledge the support obtained from Airbus Defence & Space, specially from Savier Open Innovation project members: José Insenser, César Castro and Gemma Blasco.

Appendix A. Supplementary data

Supplementary data related to this article can be found at https://doi.org/10.1016/j.swevo.2018.06.005.

References

[1] S. Perez-Carabaza, E. Besada-Portas, J.A. Lopez-Orozco, J.M. de la Cruz, A real world multi-uav evolutionary planner for minimum time target detection, in: 2016 on Genetic and Evolutionary Computation Conference - GECCO '16, ACM, 2016, pp. 981–988.
[2] V. Rodríguez-Fernández, H.D. Menéndez, D. Camacho, Analysing temporal performance profiles of UAV operators using time series clustering, Expert Syst. Appl. 70 (2017) 103–118.
[3] J. Wu, G. Zhou, High-resolution planimetric mapping from UAV video for quick-response to natural disaster, in: IEEE International Conference on Geoscience and Remote Sensing Symposium, 2006. (IGARSS 2006), IEEE, 2006, pp. 3333–3336.
[4] M.N. Bilbao, J. Del Ser, S. Salcedo-Sanz, C. Casanova-Mateo, On the application of multi-objective harmony search heuristics to the predictive deployment of firefighting aircrafts: a realistic case study, Int. J. Bio-Inspired Comput. 7 (5) (2015) 270–284.
[5] C.C. Murray, A.G. Chu, The flying sidekick traveling salesman problem: optimization of drone-assisted parcel delivery, Transport. Res. Part C Emerg. Technol. 54 (2015) 86–109.
[6] M. Mozaffari, W. Saad, M. Bennis, M. Debbah, Drone small cells in the clouds: design, deployment and performance analysis, in: Global Communications Conference (GLOBECOM), 2015 IEEE, IEEE, 2015, pp. 1–6.
[7] Y. Zeng, R. Zhang, T.J. Lim, Wireless communications with unmanned aerial vehicles: opportunities and challenges, IEEE Commun. Mag. 54 (5) (2016) 36–42.
[8] K.P. Valavanis, G.J. Vachtsevanos, Handbook of Unmanned Aerial Vehicles, Springer Publishing Company, 2014. Incorporated.
[9] S.S. Ponda, L.B. Johnson, A. Geramifard, J.P. How, Cooperative Mission Planning for Multi-uav Teams, in: Handbook of Unmanned Aerial Vehicles, Springer, 2015, pp. 1447–1490.
[10] P. Fabiani, V. Fuertes, A. Piquereau, R. Mampey, F. Teichteil-Königsbuch, Autonomous flight and navigation of VTOL UAVs: from autonomy demonstrations to out-of-sight flights, Aero. Sci. Technol. 11 (2–3) (2007) 183–193.
[11] S. Leary, M. Deittert, J. Bookless, Constrained UAV mission planning: a comparison of approaches, in: IEEE International Conference on Computer Vision Workshops (ICCV Workshops), IEEE, 2011, pp. 2002–2009.
[12] Z. Wang, Q. Liu, H. Tao, J. Li, Multiple task planning based on TS algorithm for multiple heterogeneous unmanned aerial vehicles, in: 2014 IEEE Chinese Guidance, Navigation and Control Conference, CGNCC 2014, IEEE, 2014, pp. 630–635.
[13] K. Shang, S. Karungaru, Z. Feng, L. Ke, K. Terada, A GA-ACO hybrid algorithm for the multi-UAV mission planning problem, in: 14th International Symposium on Communications and Information Technologies, ISCIT 2014, IEEE, 2014, pp. 243–248.
[14] B. Bethke, M. Valenti, J.P. How, UAV task assignment, IEEE Robot. Autom. Mag. 15 (2008) 39–44.
[15] J. Kvarnström, P. Doherty, Automated planning for collaborative UAV systems, in: 11th International Conference on Control, Automation, Robotics and Vision, ICARCV 2010, no, December, 2010, pp. 1078–1085.
[16] D. Pascarella, S. Venticinque, R. Aversa, M. Mattei, L. Blasi, Parallel and distributed computing for uavs trajectory planning, J. Ambient Intell. Humaniz. Comput. 6 (6) (2015) 773–782.
[17] R. Barták, M.A. Salido, Constraint satisfaction for planning and scheduling problems, Constraints 16 (3) (2011) 223–227.
[18] F. Guerriero, R. Surace, V. Loscri, E. Natalizio, A multi-objective approach for unmanned aerial vehicle routing problem with soft time windows constraints, Appl. Math. Model. 38 (3) (2014) 839–852.
[19] C. Ramirez-Atencia, G. Bello-Orgaz, M.D. R-Moreno, D. Camacho, Solving complex multi-UAV mission planning problems using multi-objective genetic algorithms, Soft Comput. (2016) 1–18.
[20] X. Hao, J. Liu, A multiagent evolutionary algorithm with direct and indirect combined representation for constraint satisfaction problems, Soft Comput. 21 (3) (2017) 781–793.
[21] B. Miloradović, B. Çürüklü, M. Ekström, A genetic mission planner for solving temporal multi-agent problems with concurrent tasks, in: Y. Tan, H. Takagi, Y. Shi, B. Niu (Eds.), Advances in Swarm Intelligence. ICSI 2017, Lecture Notes in Computer Science, vol. 10386, Springer Cham, 2017, pp. 481–493.
[22] E. Besada-Portas, L. De La Torre, A. Moreno, J.L. Risco-Martín, On the performance comparison of multi-objective evolutionary uav path planners, Inf. Sci. 238 (2013) 111–125.

[23] J.F. Allen, Maintaining knowledge about temporal intervals, Commun. ACM 26 (11) (1983) 832–843.
[24] K. Daniel, A. Nash, S. Koenig, A. Felner, Theta*: any-angle path planning on grids, J. Artif. Intell. Res. 39 (2010) 533–579.
[25] E. Schwalb, L. Vila, Temporal constraints: a survey, Constraints 3 (2) (1998) 129–149.
[26] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Trans. Evol. Comput. 6 (2) (2002) 182–197.
[27] C. Ramirez-Atencia, M.D. R-Moreno, D. Camacho, Handling swarm of UAVs based on evolutionary multi-objective optimization, Progr. Artif. Intell. (2017) 1–12.
[28] C. Ramirez-Atencia, G. Bello-Orgaz, M.D. R-Moreno, D. Camacho, Performance evaluation of multi-uav cooperative mission planning models, in: D.C.M. Núñez, N. Nguyen, B. Trawiński (Eds.), Computational Collective Intelligence, Lecture Notes in Computer Science, vol. 9330, Springer International Publishing, Cham, 2015, pp. 203–212.
[29] S. Iqbal, M. Kaykobad, M.S. Rahman, Solving the multi-objective vehicle routing problem with soft time windows with the help of bees, Swarm Evolut. Comput. 24 (2015) 50–64.
[30] K.V. Narasimha, E. Kivelevitch, B. Sharma, M. Kumar, An ant colony optimization technique for solving min–max multi-depot vehicle routing problem, Swarm Evolut. Comput. 13 (2013) 63–73.
[31] E. Zitzler, D. Brockhoff, L. Thiele, The hypervolume indicator revisited: on the design of pareto-compliant indicators via weighted integration, in: S. Obayashi, K. Deb, C. Poloni, T. Hiroyasu, T. Murata (Eds.), Evolutionary Multi-criterion Optimization. EMO 2007, Lecture Notes in Computer Science, vol. 4403, Springer Berlin Heidelberg, 2007, pp. 862–876.
[32] M. Hollander, D.A. Wolfe, E. Chicken, Nonparametric Statistical Methods, third ed., John Wiley & Sons, Inc, 2014, https://doi.org/10.1002/9781119196037.
[33] M. Li, L. Zhen, X. Yao, How to Read Many-Objective Solution Sets in Parallel Coordinates, 2017, pp. 1–18.
[34] T. Tusar, B. Filipic, Visualization of pareto front approximations in evolutionary multiobjective optimization: a critical review and the prosection method, IEEE Trans. Evol. Comput. 19 (2) (2015) 225–245.