A hyper-heuristic methodology for coordinating swarms of robots in target search

Mario G.C.A. Cimino [a,*], Domenico Minici [a,b], Manilo Monaco [a,b], Stefano Petrocchi [a], Gigliola Vaglini [a]

[a] *Department of Information Engineering, University of Pisa, Largo L. Lazzarino, 1, Pisa 56122, Italy*
[b] *Department of Information Engineering, University of Florence, Via di Santa Marta, 3, Florence 350139, Italy*

ARTICLE INFO

Keywords:
Target search
Swarm robotics
Bio-inspired heuristics
Evolutionary optimization

ABSTRACT

Target search aims to discover elements of various complexity in a physical environment, by minimizing the overall discovery time. Different swarm intelligence algorithms have been proposed in the literature, inspired by biological species. Despite the success of bio-inspired techniques (bio-heuristics), there are relevant algorithm selection and parameterization costs associated with every new type of mission and with new instances of known missions. In this paper, evolutionary optimization is proposed for achieving significant improvements of the mission performance. Although adaptive, the logic of bio-heuristics is nevertheless constrained by models of biological species. To generate more adaptable logics, a novel design approach based on hyper-heuristics is proposed, in which the differential evolution optimizes the aggregation and tuning of modular heuristics for a given application domain. A modeling and optimization testbed has been developed and publicly released. Experimental results on real-world scenarios show that the hyper-heuristics based on stigmergy and flocking significantly outperform the adaptive bio-heuristics.

1. Introduction and motivation

Multi-robot systems have a great potential in a variety of critical missions, such as surveillance, environmental monitoring, search and rescue. A relevant issue is the coordination of swarms for an increasing number of robots in order to achieve pre-defined global objectives. Regarding this issue, today biological swarms are much more effective with respect to the artificial counterpart. Considering that in complex or open environments robots cannot exploit static information on layout and targets locations, their cooperation is fundamental for an efficient target discovery: the key problem is how to specify the individual robot behavior for an effective interaction at the swarm level.

A target search mission is usually organized into *environmental exploration*, i.e., to search targets, and *targets resolution*, i.e., to collect sufficient target information. In the literature of biological models, a fundamental strategy of exploration is carried by ants. While on the move, ants deposit in the terrain a chemical substance called pheromone. As an example, in Ant Colony Optimization (ACO) [1] artificial ants release pheromones while exploring the environment to temporarily mark the visited places. Different types of

* Corresponding author.
E-mail address: mario.cimino@unipi.it (M.G.C.A. Cimino).

pheromone, related to diverse meanings, enable ants to make different decisions. Digital versions of pheromone are commonly used to orientate robots' exploration [2]. Robots move according to the sensed pheromone; specifically, a robot begins to coordinate the resolution of the target detected during exploration, by attracting other robots towards the position indicated by the pheromone. When the recruited robots are sufficient in number, they perform the target resolution. In the literature, three major bio-inspired meta-heuristics are considered for recruitment: (i) the Firefly-based Team Strategy (FTS) [3], an algorithm derived from swarms of fireflies; (ii) Particle Swarm Optimization (PSO), modelled from schools of fish and flocks of birds; (iii) Artificial Bee Colony (ABC), based on honey bees [4]. The problem of coordinating swarms of robots has received attention by many research areas, due to its potential impact on real-world applications. Swarm coordination strategies can be divided into two categories. *Explicit* coordination is based on the direct exchange of messages between robots, according to a detailed orchestration among swarm members [5]. This many-to-many communication strategy causes a poor performance of large swarms of robots. In contrast, with *implicit* coordination each robot makes simple behavioral decisions, based on information gathered through its indirect perception mediated by an environmental mechanism. Although the single piece of information obtained by perception is not completely accurate, the robustness of the swarm can be sensibly improved by the collective contribution [5].

Despite the success of bio-inspired techniques (bio-heuristics), there are relevant algorithm selection and parameterization costs associated with every new type of mission and with new instances of known missions. In this paper an evolutionary optimization is proposed to automate the tuning of the bio-inspired coordination for target search. Experimental results on real-world scenarios reveal a significant improvement of the mission performance after optimization.

Although adaptive, the logic of bio-heuristics is nevertheless constrained by models of biological species, and then, for example, it can be neither modularised nor aggregated. To overcome these limits, a novel design approach based on *hyper-heuristics* (HH) is proposed. It is a search methodology that automates the combination of modular heuristics to generate more adaptable logics: fundamental behavioral components for many biological swarms are aggregated and tuned in a unique and continuous search space. Two fundamental swarm behavioral components are considered: *stigmergy* and *flocking*. Stigmergy is used to release an attractive – or repulsive – stimulus when detecting the presence/absence of a target during exploration. Multiple stimuli can overlap, creating a stigmergic trail which, in turn, evaporates over time. As a result, stigmergy creates a kind of context-aware memory of the swarm [6]. Flocking is used to model a robust and flexible swarm formation. It is based on the rules of cohesion, separation and alignment [2]. Depending on the type of mission and on the environment layout, flocking of different sizes and flexibility can be adaptively modelled.

The *Differential Evolution* (DE) algorithm optimizes the aggregation and tuning of the heuristics on a unique search space and, consequently, an efficient heuristics hybridization is generated for a given application domain. DE is a population-based metaheuristic optimization algorithm, based on computational mechanisms of biological evolution, such as reproduction, mutation, recombination, and selection of solutions. DE can tackle non-linear and complex optimization problems, requiring just the objective function values. A modeling and optimization testbed has been developed and publicly released [7]. Experimental results on real-world scenarios show that the proposed approach, called SFE because it is based on Stigmergy, Flocking, and Evolution, significantly outperforms the adaptive bio-heuristics.

The paper is structured as follows. Section 2 covers the related work. In Section 3 the design of the proposed solutions is formally discussed. Testbed and scenarios are detailed in Section 4. Section 5 is devoted to experimental setup and results. Finally, in Section 6 the conclusions are drawn.

2. Related work

2.1. Bio-inspired swarm robotics

In the literature, many algorithms proposed for modeling swarms of robots are inspired by biological systems. In particular, solutions based on insect colonies manifest interesting properties such as local control and communication, self-organization and emergence of global behavior [1, 3]. For instance, ants release attractive pheromone trails as a medium for self-organization to mark and reinforce their most frequent paths. Coordination strategies for robots based on chemical trails have been experimented; for instance, in Fujisawa *et al.* [8], ethanol trails have been used by robots as a medium to deposit and follow. However, chemical trails can cause problems for environmental impact, maintenance costs, control of transmission speed and range. For this reason, non-chemical media are more effective [9]. Masár *et al.* [10] have proposed a variant of the PSO algorithm for environment exploration. Another significant bio-inspired algorithm is called Artificial Bee Colony (ABC): it is based on the food foraging process of honey bees. ABC variants have been used to coordinate robotic systems [11]. Another well-known algorithm is the Firefly-based Team Strategy (FTS), proposed by Yang [3] and based on the flashing behavior of fireflies. In this paper FTS, PSO and ABC will be considered as a reference for swarm coordination in target search missions. For an extensive review, the interested reader can refer to [5, 12].

Bio-inspired techniques, albeit provided with a certain variety of approaches, are still not organized as an operational framework: design and setting costs associated with every new type of mission and with new instances of known missions are a major drawback. The major difficulties are due to the lack of reliable guidance on how to select the algorithms and the parameters in different situations. When applied to real-world problems, the method tends to become bespoke and problem-specific, characterized by expensive development and maintenance. For this purpose, a promising research direction is called *algorithm configuration*, whose goal is to automatically determine the appropriate parameters values for an algorithm. The goal can be considered as a search problem in the configuration space, in which the objective function measures the algorithm performance over a benchmark [13]. Another approach is called *parameter control*, which performs an online tuning of the algorithm parameters at execution time [14]. To automate the tuning of bio-inspired methods for target search is a challenge in the field [15]. For this purpose, in this paper evolutionary optimization is

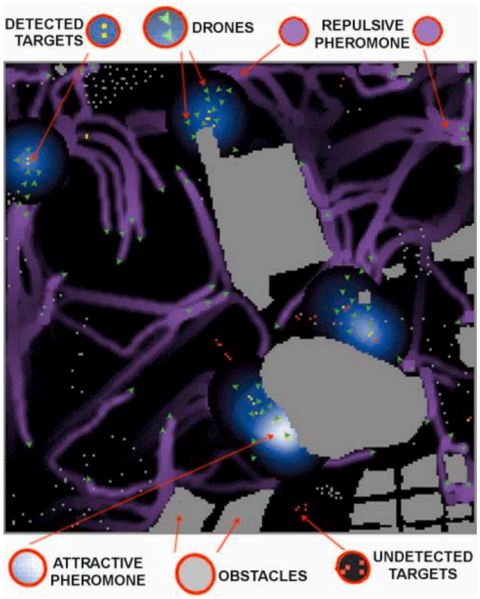

Fig. 1. Environment: drones, targets, attractive and repulsive pheromone, obstacles.

adopted.

2.2. Hyper-heuristics

Although adaptive, the logic of bio-heuristics is nevertheless constrained by models of biological species and can be neither modularized nor aggregated. To overcome these limits, a novel design approach based on hyper-heuristics (HH) is proposed in this paper. It is a search method that automates the combination of modular heuristics to generate more adaptable logics. In [16] Burke *et al.* have presented a unified classification and definition of HH able to capture the research work in the field. They define HH as a search method or a learning mechanism to select or generate heuristics solving search problems. Specifically, in a *learning* HH a feedback is given from the search process. In *online* learning HH the learning occurs while solving the problem, whereas in *offline* learning HH knowledge is gathered from training instances and modelled as rules or programs. Considering the type of search space, the *heuristic selection* chooses or selects predefined heuristics, whereas the *heuristic generation* generates new heuristics from modular components. Both search paradigms can be further divided into *perturbative*, when adjusting full candidate solutions by modifying their components, and *constructive*, when iteratively extending partial candidate solutions with missing components. Hybrid approaches are also used [17]. In particular, Garrido *et al.* [17] have solved the dynamic vehicle routing problem via an evolutionary HH. Their framework is based on a combination of both constructive and perturbative HH, and is evaluated on a large and complex set of problems. Results are competitive with respect to well-known methods of the literature. The HH approach aims to provide a general method for many application domains, rather than a better solution to a specific problem. Indeed, the search space of HH is a space of new heuristics, rather than a space of solutions. The difference is that a new heuristic can be potentially reused for solving many problem instances. In the literature, a well-known method for generating heuristics is genetic programming [18]. It is an evolutionary computation evolving a population of computer programs. Genetic programming can be considered as an HH if the evolved programs are heuristics. For example, in [19] Geiger *et al.* have illustrated the major motivations to automatically generate heuristics in production scheduling. The research in the field has shown also that successful components can be derived by the available human-created heuristics [20]. Another research field, related to perturbative heuristics, is called *adaptive memetic algorithms*. It is an evolutionary algorithm characterized by local searchers called memes, adaptively selected/generated during the search [21, 22].

2.3. Evolutionary algorithms

Evolutionary algorithms (EAs) are population-based metaheuristic optimization algorithms, based on computational mechanisms of biological evolution, such as reproduction, mutation, recombination, and selection of solutions. EAs can tackle non-linear and complex optimization problems, requiring just the objective function values. Nevertheless, the performance of an EA depends, in turn, on hyper-parameter settings: for instance, probabilities of mutation and crossover, population size, and number of generations. Parameter tuning methods have been proposed, such as deterministic, adaptive and self-adaptive [22]. Here, the Differential Evolution (DE) algorithm has been considered: it is an evolutionary algorithm for optimization in continuous spaces. The performance of the DE depends on the mutation control parameters, especially when the problem is complex [23]. To balance the convergence (fitness evaluations) and the reliability (optimum's globality), ranges of parameters values have been studied. The most popular variant of DE

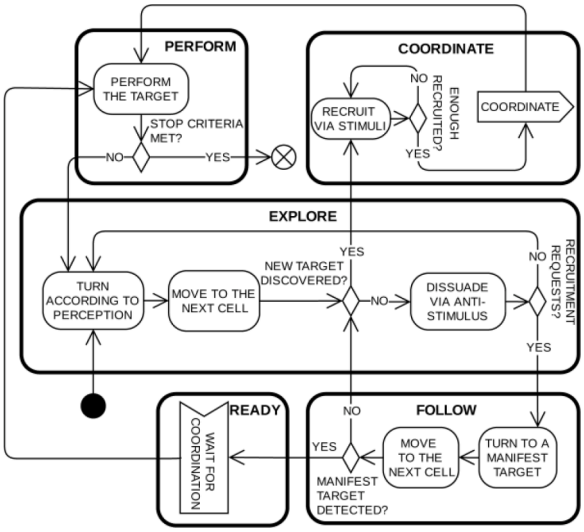

Fig. 2. Target search mission: UML activity diagram with the overall workflow

is called "DE/rand/1/bin", where "DE" stands for Differential Evolution, "rand" means that the individuals selected to compute the mutation values are randomly chosen, "1" is the number of pairs of individuals chosen for mutation, and "bin" denotes the binomial crossover. Another variant is based on the best selection strategy: it is called "DE/best/1/bin", because the perturbing individual is generated from the best population member. It is known that "DE/rand/1/bin" is slow but robust compared to the strategies based on the best member. Among the most sensitive parameters, the Crossover Rate (*CR*) is a probability of mixing between mutant (donor) and target vectors of the current population [24]. Low/high *CR* values are good for uni/multi-modal problems. Good convergence can be achieved with large *CR* values. Recommended *CR* values are in [.2, .9]. The differential weight, $F \in [0, 2]$ controls the mutant vector: large *F* values allow escaping from local optima; low values cause premature convergence; $F \leq 1$ determines a fast and reliable search process. As a result, *F* is usually set in [0.4, 0.9]. The population size (*NP*) is another important parameter. In the literature, there is a lack of sufficient justifications and a lot of conflicting motivations about the manual parameter tuning of DE. To solve the issue, in this research a grid search technique is used.

3. From adaptive bio-heuristics to hyper-heuristics

In this section, the exploration and recruitment problems are formalized. The FTS, PSO and ABC bio-heuristics are then defined, together with their user-defined parameters. Subsequently, the adaptive variant of the bio-heuristics is presented. Finally, a different solution, based on hyper-heuristics, is proposed.

3.1. Environment and problem statement

Fig. 1 shows an ongoing scenario of target search. Let us consider a swarm of mobile robots, or drones, deployed in an exploration area, in order to search and process the targets cooperatively. Let us assume that targets number and locations are unknown. In Fig. 1, obstacles are represented in grey color. Drones are depicted as green arrowheads, and undetected/detected targets as red/yellow points. Finally, an attractive/repulsive pheromone is represented as white/pink continuous intensity. The swarm is also divided into flocks: flexible, dynamic and autonomous groups communicating between themselves (flockmates) and self-organizing, splitting around obstacles, rejoining, and avoiding collisions with each other. Moreover, an attractive pheromone released by flockmates creates a short-medium term potential to compact the flock where multiple targets are detected. In contrast, a repulsive pheromone helps the drones to avoid multiple exploration of the same zone whereas new targets are not detected. Finally, olfactory habituation is another bio-inspired form of memory: when exposed to the maximum intensity of attractive pheromone, the sensing saturates and becomes unable to sense for a while, to leave the saturated area more efficiently.

More formally, let be:

- $W \subset \mathbb{R}^2$, exploration area; in the computerized model it is actually a discretized area in $\mathbb{N}^2$;
- *R*, set of robots of the autonomous swarm, with $N^R = |R|$;
- N^R_{min}, number of robots needed to process each target;
- *RR*, set of robots recruited to process a target, $RR \subset R$;
- *T*, set of all targets, with $N^T = |T|$;
- *F*, set of found targets, $F \subset T$, with $N^F = |F|$;
- *FP*, set of targets found and performed, $FP \subseteq F$, $N^{FP} = |FP|$;

- FHR_k, set of help requests, i.e., found targets received by the k-th robot, $FHR_k \subset F \subset T$.

In a target search mission, a robot can assume two major roles: explorer and coordinator [25]. The purpose of exploration is to discover new targets, whereas the purpose of coordination is to recruit the necessary number of follower robots to process the discovered target. Fig. 2 shows a UML activity diagram with the overall workflow carried out by each robot involved in a target search mission. The process begins at the black start circle, and ends at the white circle with a cross inside. The major activities are represented by bold round-cornered rectangles, and are connected by the following two core flows:

1) *explore* → *coordinate* → *perform*
2) *explore* ⇄ *follow* → *ready* → *perform*

Each activity is defined as a work flow of generic tasks. The implementation of a task can vary depending on the bio-inspired approach (e.g., ACO for exploration; FTS, PSO, and ABC, for recruitment [25]).

It follows the description of each activity. *Explore*: the robot explores the area for discovering targets; first, it is oriented by its perception of a medium depending on the biological model; then, it moves to the next cell; if a new target is discovered, it *coordinates*; otherwise it dissuades from following its recent path by releasing some anti-stimulus; finally, if no recruitment request arrives, the robot continues to *explore*, otherwise it *follows*. *Follow*: the robot is recruited by a coordinator robot; it selects one of the manifest targets, then turns to it, and moves to the next cell; when the manifest target is detected, the drone waits for coordination (*ready*). *Coordinate*: the robot becomes a coordinator when it detects a target, and after it starts to recruit the needed robots; the recruitment is based on stimuli depending on the bio-inspired approach. *Ready*: once reached the target, a recruited robot waits until the coordinator delivers the authorization to *perform* the target. *Perform*: the target is processed by all recruited robots; then, a stop criterion is checked, e.g. a maximum time or a maximum percentage of targets found.

The next sections will formalize the major tasks for the reference bio-inspired approaches [25].

3.2. Exploration based on the ACO bio-heuristic

During the *explore* activity, a repulsive pheromone is deposited on the visited cells without targets, as an anti-stimulus, in order to temporarily mark them. The structure and dynamics of the repulsive pheromone model is derived from ACO [1, 25]. Formally, the amount of pheromone deposited by the k-robot located at $(x_k^t, y_k^t) \in W$ at iteration t, on the cell c located in (x, y) is given by:

$$\Delta\varphi_{k,c}^t = \begin{cases} \Delta\varphi_0 e^{\frac{-r_{kc}}{a_1}} - \dfrac{\varepsilon}{a_2} \;\; if \;\; r_{kc} \le r_P \\ 0 \;\; otherwise \end{cases} \tag{1}$$

where r_{kc} is the Euclidean distance of the k-th robot from the cell c, $\Delta\varphi_0$ is the maximum quantity of pheromone, released on the robot location, $\varepsilon \in (0, 1)$ is a heuristics-based value, and a_1 and a_2 are two sizing constants. The quantities of pheromone released by many robots in a cell c are summed. The pheromone quantity that evaporates in the cell c at step t is:

$$\xi_c^t = \rho\varphi_c^t \tag{2}$$

where φ_c^t is the pheromone quantity on the cell c at iteration t, whereas ρ is the rate of evaporation:

$$\rho = (t - t_v)ERTU_{\%} \tag{3}$$

in which t_v is the last time the cell was visited, t is the current time, and $ERTU_{\%}$ (Evaporation Rate Time Unit) is the coefficient of evaporation rate per unit of time spent.

Given the pheromone evaporation over time and the diffusion of it over the distance, the pheromone amount in the cell c at time t is:

$$\varphi_c^t = \varphi_c^{t-1} - \xi_c^{t-1} + \sum_{k=1}^{N^R} \Delta\varphi_{k,c}^t \tag{4}$$

Each k-th exploring robot located at cell c_k^t perceives the pheromone released into the neighbor cells $N(c_k^t)$, and has a probability to move to cell $c \in N(c_k^t)$:

$$p\left(c|c_k^t\right) = \frac{\left(\varphi_c^t\right)^{\psi}\left(\eta_c^t\right)^{\lambda}}{\sum_{b \in N\left(c_k^t\right)}\left(\varphi_b^t\right)^{\psi}\left(\eta_b^t\right)^{\lambda}}, \;\; \forall c \in N\left(c_k^t\right) \tag{5}$$

where $(\varphi_c^t)^{\psi}$ is the pheromone level in cell c at time t, $(\eta_c^t)^{\lambda}$ is a heuristic variable, ψ and λ are two constant quantities. According to the exploration based on ACO, the k-robot moves to the cell c determined by:

$$c = \min\left[p\left(c|c_k^t\right)\right] \tag{6}$$

in other words, it moves to the less frequented or unexplored areas.

3.3. Coordination based on the FTS bio-heuristic

FTS is modelled on the flashing-based coordination of fireflies, which communicate variating light intensity and attractiveness [3]. The firefly brightness is modelled by an object function, whereas its attractiveness is proportional to the brightness. More formally, given two fireflies i and j, located at x_i and x_j, respectively, the attractiveness function β of the firefly j is:

$$\beta = \beta_0 e^{-\gamma d_{ij}^2} \tag{7}$$

where d_{ij} is the Euclidean distance, β_0 is the maximum attractiveness, and γ is an absorption coefficient. A firefly i which is attracted by a brighter firefly j moves as follows:

$$x_i^{t+1} = x_i^t + \beta\left(x_j^t - x_i^t\right) + \alpha\left(\sigma - \frac{1}{2}\right) \tag{8}$$

where on the right there is a randomization term: α is the randomization parameter and σ is a scaling factor, usually set to 1. Good parameter settings are needed for achieving good convergence and stability [3].

Attraction begins when a robot becomes a coordinator: the coordinated robot x_i move to target (coordinator x_j) according to (8). If subject to many requests, the robot follows the brighter target located in a threshold distance. If more targets are found, the randomization term in (8) avoids that the recruited robots move to the same target. A further constraint for being coordinated is $d_{ij} \leq (r_W + \Delta)$, where r_W is the communication range between robots, and Δ is a perturbation coefficient. Thus, the robot i switches to the exploration task if it becomes too far from its target j.

3.4. Coordination based on the PSO bio-heuristic

PSO is an optimization technique modelled on flocks of birds. For each i-th "particle", moving with speed v_i^t and position x_i^t in the search space, the new position is calculated by:

$$x_i^{t+1} = x_i^t + v_i^{t+1} \tag{9}$$

where v_i^{t+1} is the speed in unit time, calculated according to the target positions x_j, over all its neighbors:

$$v_i^{t+1} = \omega v_i^t + r\ c\ \left(x_j - x_i^t\right) \tag{10}$$

where r is a uniform random number in [0, 1], ω is the inertial weight, and c is an acceleration coefficient. In case of many recruiting requests, the robot moves to the closest target.

3.5. Coordination based on the ABC bio-heuristic

The ABC algorithm is modelled on the foraging process of honey bees. There is a population of food positions, and bees that seek, advertise, and select the best food position. A solution (food source) x_i is a D-dimensional vector, whose fitness value is associated with its position. There are BN onlooker bees, BN employed bees, and SN food sources, with $SN{=}BN$. An onlooker bee selects the food source according to the following probability of fitness p_i:

$$p_i = \frac{fit_i}{\sum_{q=1}^{SN} fit_q} \tag{11}$$

where fit_i is the fitness of the i-th solution, provided by its employed bee. According to an iteration logic, the next candidate food position is derived from the previous one:

$$x_i^{t+1} = x_i^t + \phi\left(x_i^t - x_j\right) \tag{12}$$

where x_i^{t+1} is the new food position, calculated from the previous one x_i^t and the selected target x_j; φ is a random number between $[-1, 1]$. In case of more than one request: the k-th robot selects the z-th target with probability:

$$p_{kz} = \frac{1/d_{kz}}{\sum_{b=1}^{FHR_k} 1/d_{kb}} \tag{13}$$

where $FHR_k \subset F \subset T$, and d_{kz} is the Euclidean distance.

3.6. Adaptive algorithm configuration and parametric space

Despite the success of bio-inspired techniques, there are relevant algorithm selection and parameterization costs associated with every new type of mission and with new instances of known missions. In this paper, the DE is adopted for the parametric adaptation of

Table 1
Parameters space of the ACO-E

Parameters	Interval
r_P (pheromone range)	[0, 8]
$ERTU_{\%}$	[0, 1]
$\Delta\varphi_0$	[0, 4]
ε	Uniform [0, 1]
a_1	[0, 2]
a_2	[0, 2]
η	[0, 2]
ψ	[0, 2]
λ	[0, 2]

Table 2
Parameters space of the ACO-FTS-RR3-E

Parameters	Interval
r_W (perception range)	[1, 19]
r_P (pheromone range)	[0, 8]
$ERTU_{\%}$	[0, 1]
$\Delta\varphi_0$	[0, 4]
ε	Uniform [0, 1]
a_1	[0, 2]
a_2	[0, 2]
η	[0, 2]
ψ	[0, 2]
λ	[0, 2]
β_0	[0, 1]
γ	$1/L(L = max\{m, n\})$
α	[0, 0.4]
σ	Uniform [0, 1]

the bio-inspired exploration and recruitment algorithms on target search. The quality measure of a target search is the time needed for completing the mission, i.e., for discovering a given percentage of target [26]. As a consequence, the fitness of the DE is defined as the mission duration. More formally, given a simulated scenario Ω, made of: (i) simulation instants of time $t \in \mathbb{N}^+$; (ii) a set of robots R, each robot k having a dynamic position (x_k^t, y_k^t); (iii) a set of targets $z \in T$, each target having a fixed position $(x, y)_z$. Hence, the set of found targets $F(t) \subseteq T$, at a given instant of time t, is the set of targets $\{z\}$ for which it exists a time $t' \leq t$ and a related set of robots $\{k_i^z\}$, $i = 1, \ldots N_{min}^R$, such that the robots' Euclidean distances from the target position is lower than the detection distance δ:

$$F(t) = \left\{ z | \ \exists \ k_i^z, i = 1, \ldots N_{min}^R, \exists t' \leq t : \ d\left[\left(x_{k_i^z}^{t'}, y_{k_i^z}^{t'} \right), (x, y)_z \right] \leq \delta \right\} \tag{14}$$

The fitness of the simulated scenario Ω is then defined as the minimum instant of time for which $F(t)$ has cardinality greater than or equal to $\vartheta \bullet |T|$:

$$fitness(\Omega) = \min_{t \in \mathbb{N}^+} \{ t : \ |F(t)| \geq \ \vartheta \cdot |T| \} \tag{15}$$

where ϑ is a percentage threshold close to 1 (usually set to .95) used to reduce the simulation duration without sensibly affecting the accuracy.

More formally, the DE logic is summarized by the pseudocode presented in Algorithm 1. In a simulated scenario (or mission), the swarm S_i explores an environment where *Robots, Obstacles* and *Targets* are statically specified. Let K be the number of aggregated parameters. In the DE, S_i is a solution represented by a real K-dimensional vector called genotype $\boldsymbol{p}_i$. The search time returned by the simulated mission is used as a fitness of the solution, f_i. DE starts with a population $P^{(0)}$, made by N candidate solutions, $\boldsymbol{p}_i^{(0)}$, randomly generated under user-specified parametric constraints. At each iteration t, and for each genotype $\boldsymbol{p}_i^{(t)}$ of the current population $P^{(t)}$, a mutant vector $\boldsymbol{m}$ is created by applying the mutation of randomly selected members. Then, a trial vector $\boldsymbol{p}_i^*$ is created by crossover of $\boldsymbol{m}$ and $\boldsymbol{p}_i^{(t)}$. Subsequently, the population is modified selecting the best fitting vector between the fitness of the trial vector (f_i^*) and the fitness of the initial genotype ($f_i^{(t)}$). When the termination criterion is true, i.e., number of iterations performed or adequate fitness reached [27], the vector characterizing the swarm with the best fitness (i.e. the shortest search time) in the current population is considered as the optimal swarm parameterization. The DE algorithm has at least two hyper-parameters: the scaling factor $F \in [0, 2]$ from which results the mutant vector, and the crossover probability CR. The smaller CR the higher the probability of producing a vector that is more similar to the target vector rather than to the mutant vector. More formally, Algorithm 2 and Algorithm 3 define the mutation and the crossover operators, respectively.

According to this approach, the DE finds the optimum in the parametric search space of the bio-inspired algorithm which, in turn,

Table 3
Parameters space of the ACO-PSO-RR3-E

Parameters	Interval
r_W (perception range)	[1, 19]
r_P (pheromone range)	[0, 8]
$ERTU_{\%}$	[0, 1]
$\Delta\varphi_0$	[0, 4]
ε	Uniform [0, 1]
a_1	[0, 2]
a_2	[0, 2]
η	[0, 2]
ψ	[0, 2]
λ	[0, 2]
ω	[0.4, 1]
r_1	Uniform [0, 1]
c_1	[0, 4]

Table 4
Parameters space of the ACO-ABC-RR3-E

Parameters	Interval
r_W (perception range)	[1, 19]
r_P (pheromone range)	[0, 8]
$ERTU_{\%}$	[0, 1]
$\Delta\varphi_0$	[0, 4]
ε	Uniform [0, 1]
a_1	[0, 2]
a_2	[0, 2]
η	[0, 2]
ψ	[0, 2]
λ	[0, 2]
φ	Uniform [-1, 1]

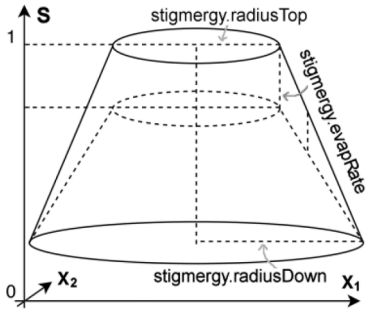

Fig. 3. Model of a stigmergic mark

solves a target discovery problem in a bidimensional space, via exploration and recruitment of robots [28]. Thus, for each bio-inspired algorithm, a corresponding adaptive variant is proposed.

Considering the ACO for exploration, the FTS, PSO and ABC for recruitment, the names of the corresponding variants are: ACO-E, FTS-E, PSO-E and ABC-E, where the term "E" means Evolution. An algorithm variant solving both exploration and recruitment includes two acronyms and is parameterized in a search space that is the union of the two search spaces. Since the recruitment problem varies significantly in complexity depending on the number of robots needed to process a target, N^R_{min}, a term "RR*" is added to highlight the different complexity. For example, "RR3" means an algorithm for recruitment with $N^R_{min} = 3$, whereas "RR1" means an algorithm without recruitment, i.e., an exploration algorithm. When both exploration and recruitment problems are considered, the name of the algorithmic solution includes both acronyms. For example, ACO-ABC-RR3-E. Table 1, Table 2, Table 3 and Table 4 show the parametric spaces of ACO-E (used only for exploration task), ACO-FTS-RR3-E, ACO-PSO-RR3-E and ACO-ABC-RR3-E, respectively.

3.7. Exploration and coordination via the SFE hyper-heuristic

In this section we consider a different algorithmic design based on hyper-heuristics. In this approach, the logic is not constrained by models of biological species. It consists of an optimization method of fundamental functional components, whose aggregation and

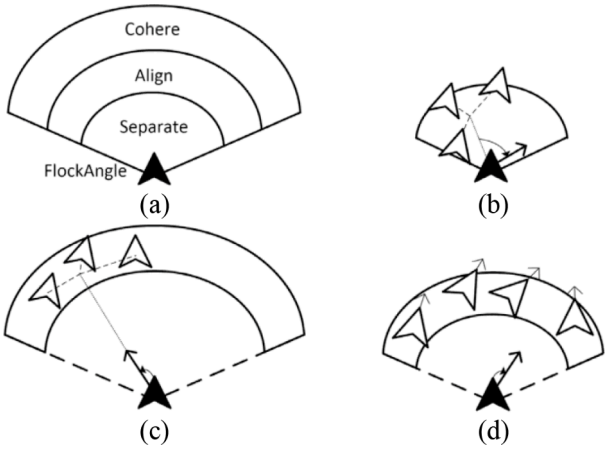

Fig. 4. Model of the flocking behavior: (a) regions, (b) separate, (c) cohere, (d) align.

Table 5
Parameters space of the SFE-RR1

Parameters	Interval
stigmergy.radiusTop	[1, 13]
stigmergy.radiusDown	[13, 19]
stigmergy.evapRate	[0.01, 1]
stigmergy.olfactoryHabituation	[1, 10]
stigmergy.repulsiveRadius	[0, 8]
stigmergy.repulsiveEvapRate	[0.01, 0.5]
flocking.angle	[15, 45]
flocking.wiggleVar	[5, 15]
flocking.radiusSeparate	[6, 16]
flocking.maxSeparateTurn	[30, 45]
flocking.radiusAlign	[16, 22]
flocking.maxAlignTurn	[30, 45]
flocking.radiusCohere	[18, 26]
flocking.maxCohereTurn	[15, 30]

tuning are represented on a unique and continuous search space. More formally, let us consider two fundamental swarm behavioral components, stigmergy and flocking [29].

Stigmergy is used to release an attractive (or repulsive) stimulus while (not) detecting targets. In the proposed computational model, a digital stigmergic mark is released by the robot in the environment. Fig. 3 illustrates the model of the stigmergic mark: it is a truncated cone with unit height, radius top and down. Multiple stigmergic marks can overlap, creating a stigmergic trail. Stigmergic trails evaporate over time. Since the stigmergic trail is maintained in a digital environment, it is instantly diffused, to immediately propagate information to nearby robots. More formally, let us consider the target z detected by the robot k at time t, with position $(x_z^t, y_z^t) \in W$. The pheromone quantity $\Delta s_{k,c}^t$ released on the cell c located in (x, y) is given by:

$$\Delta s_{k,c}^t = \begin{cases} 1 & if\ d_{zc} \le r_{top} \\ \dfrac{d_{zc} - r_{down}}{r_{top} - r_{down}} & if\ r_{top} < d_{zc} < r_{down} \\ 0 & otherwise \end{cases} \tag{16}$$

where $r_{top} and r_{down}$ are *radiusTop* and *radiusDown*, respectively, d_{zc} is the Euclidean distance between the target z and the cell c. The trail intensity in the cell c at time t is given by:

$$s_c^t = \max\left\{0, \min\left\{s_{max}\ ,\ s_c^{t-1} - e_{rate} \cdot s_c^{t_r} + \sum_{k=1}^{N^R} \Delta s_{k,c}^t\right\}\right\} \tag{17}$$

where $s_c^{t_r}$ is the trail intensity on the cell c at the instant of the last pheromone release t_r, and e_{rate} is *evapRate*, i.e., the given amount of intensity evaporated per unit time. The model with a linear evaporation and a streamlined shape allows a good control of the aggregated trail in the parameter space. The perceived stigmergic intensity is based on olfactory receptors, which can decrease in sensibility over time to prevent overstimulation (olfactory habituation).

Table 6
Parameters space of the SFE-RR3

Parameters	Interval
stigmergy.radiusTop	[1, 13]
stigmergy.radiusDown	[13, 19]
stigmergy.evapRate	[0.01, 1]
stigmergy.olfactoryHabituation	[1, 10]
stigmergy.repulsiveRadius	[0, 8]
stigmergy.repulsiveEvapRate	[0.01, 0.5]
flocking.angle	[15, 45]
flocking.wiggleVar	[5, 15]
flocking.radiusSeparate	[6, 16]
flocking.maxSeparateTurn	[30, 45]
flocking.radiusAlign	[16, 22]
flocking.maxAlignTurn	[30, 45]
flocking.radiusCohere	[18, 26]
flocking.maxCohereTurn	[15, 30]

Flocking is used to model a robust and flexible swarm formation. It is based on the rules of cohesion, separation and alignment, illustrated in Figure 4. The different rules are activated in separate regions (Figure 4a). The separation rules (Figure 4b) maintain a distance among flock mates for a better scan of the area. The cohesion rules (Figure 4c) direct the robot to the flock center, to avoid dispersion. Finally, the alignment rules (Figure 4d) keep the heading of each robot aligned to the average heading of its flock mates. Depending on the type of mission, flocking of different sizes can be modelled.

An efficient heuristics hybridization is generated for a given application domain, in which DE minimizes the mission discovery time. The resulting algorithm is called SFE (Stigmergy, Flocking, Evolution) [26]. Since SFE can adapt his behavior to the problem, it can be used for both exploration and recruitment. More specifically, considering Figure 2: (i) a repulsive stigmergy is used as an anti-stimulus during the exploration (ii) an attractive stigmergy is used as a stimulus for recruitment; (iii) during exploration, the robot turns primarily to the maximum attractive pheromone, if detected, else follows the flocking rules, if flockmates are detected;otherwise it turns to the minimum repulsive pheromone.

Table 5 and Table 6 show the parametric spaces of SFE-RR1, for exploration, and SFE-RR3 for both exploration and recruitment. Note that when the parameters *radiusTop, radiusDown*, and *repulsiveRadius* are very small, then attractive and repulsive stigmergy are very low too. Similarly, if *flocking.angle* in Figure 4a is very small, then flocking is very low since no flockmate is visible. Thus, such parameters can lower/raise the contribution of each component in the overall workflow, in a continuous optimization space.

3.8. Management of the stochastic behavior

An important aspect to consider is the control of the uncertainty potentially resulting from the initial swarm position and from the random-evaluated parameters. For this purpose, the initial swarm position is fixed: the swarms are initially located at the corners of the environment and oriented towards the center of it. However, there are two sources of non-determinism that can further influence the algorithmic performance.

The first source occurs at the application level of the target search, because all swarm algorithms inherently include random-valued parameters: *wiggle* (SFE), ε (ACO), σ (FTS), r_1 (PSO), and ϕ (ABC). To manage this uncertainty, we adopt confidence intervals as a way to measure performance beyond statistical fluctuations. Furthermore, in contrast to the other swarm algorithms, the SFE allows to adapt the range of the *wiggle* via the DE optimization, for achieving the best cost-uncertainty ratio.

The second source of non-determinism occurs at the optimization level provided by the DE. Specifically, the *intializePopulation* function is managed via the lower/upper bounds per parameter, and by a Latin Hypercube sampling to maximize the coverage of the available parameter space. The *generateMutant* also involves multiple random extractions, except for the DE/best/1/bin, to select the best individual as a base vector $\boldsymbol{p}_i^{(t)}$. Finally, the *binomialCrossover* includes some random extractions, managed by the parameter *CR*. To control the last two variabilities, two mutation strategies and various *CR* values have been compared in the hyperparameters search. Finally, to further reduce the overall uncertainty, each fitness evaluation is measured as an average of 10 trials, and the best result provided by the DE is calculated as an average of 3 independent trials made by 40 generations.

4. Testbed and scenarios

A modeling and optimization testbed has been publicly released [7]. In addition to environment and swarm algorithms, the testbed considers the robots sensing, actuation, and collision avoidance, by modeling drone size, battery duration, sensing radius, sensing angle, collision angle, collision vision angular speed, acceleration, and cruise speed. Three scenarios of different complexity have been considered. The *Illegal Dump* scenario represents a real Abusive Trash Map of 80,000 m^2 near the town of Paternò (Italy), and is composed of 11 groups of targets with an average number of 4 targets per group, 19 buildings of different sizes, 140 trees (www.trashout.me). Figure 5 and Figure 6 show the aerial photo and the corresponding vectorial model, respectively.

The *Rural Mine* scenario is a real-world example of areas with landmine objects in Bosnia-Herzegovina, described in public data (www.seedemining.org). It is made up of 28 buildings, 59 trees and 40 targets. Figure 7 and Figure 8 show the aerial photo and the

Fig. 5. Illegal Dump scenario: aerial photo (Google Maps ©)

Fig. 6. Illegal Dump scenario: vectorial model

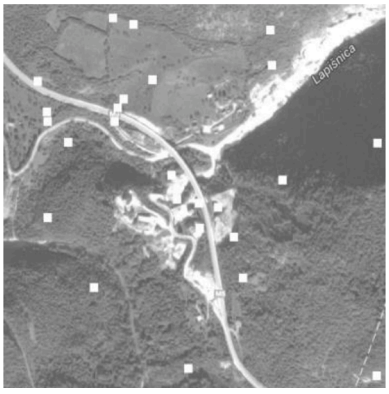

Fig. 7. Rural Mine scenario: aerial photo (Google Maps ©)

corresponding vectorial model, respectively. The *LPG Leak* scenario, is based on an accident caused by an LPG railcar rupture, which occurred in 2009 in the urban area of Viareggio, Italy [30]. Figure 9 and Figure 10 show the aerial map [30] and the corresponding vectorial model.

5. Experimental setup and results

The most sensitive hyper-parameters of DE are the differential weight (*F*), the crossover rate (*CR*), and the population size (*NP*). We

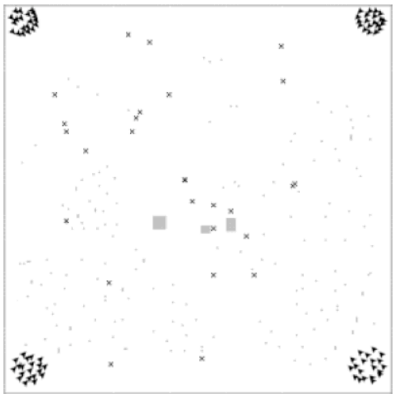

Fig. 8. Rural Mine scenario: vectorial model

Fig. 9. LPG Leak scenario: aerial map [30]

Fig. 10. LPG Leak scenario: vectorial model

use a multiplier of the problem dimension for setting the total population size: the population has $4D$ individuals. Based on the literature, as discussed in a previous section, the range of values to consider are CR in [.1, .9] with steps of 0.1, F in [.4, .9] with steps of 0.1. Two mutation strategies have been experimented. Figure 11 and Figure 12 show the grid search on the *Illegal Dump* scenario, with the ACO-E algorithm, for the DE/rand/1/bin ("r" for short) and the DE/best/1/bin ("b" for short), respectively. Here, the minimum duration of 189.3 $\pm$ 26.55 (r) and 190.0 $\pm$ 20.46 (b) is achieved for (CR, F) equals to (.8, .4) (r) and (.7, .4) (b), respectively. In both figures, the optimal values are highlighted with a small circle in the (CR, F) plane.

Similarly, Figure 13 and Figure 14 show the grid search process with the SFE-RR1 algorithm, for DE/rand/1/bin (r) and DE/best/1/bin (b), respectively. Here, the minimum duration of 104.8 $\pm$ 10.45 (r) and 121.0 $\pm$ 02.99 (b) is achieved for (CR, F) = (.4, .4) (r) and (.6, .5) (b), respectively. As a result, the DE/rand/1/bin strategy achieves better performance than DE/best/1/bin with the SFE-RR1 algorithm, and achieves performance similar to DE/best/1/bin with the ACO-E algorithm. Overall, the effectiveness of DE/rand/1/bin can be considered better.

By using the DE/rand/1/bin and the optimal values of hyperparameters determined by the grid search, a comparative analysis of the different algorithms is carried out. For each scenario and for each strategy, the DE optimization is carried out 10 times, determining via a graphical normality test that the resulting mission duration is well modelled by a normal distribution. Finally, the 95% confidence intervals are calculated. Table 7 and Table 8 show the mission duration before and after the DE. Here, it is apparent that the swarm exploration and recruitment carried out by the proposed SFE outperform the other strategies. Furthermore, it is clear that the DE optimization sensibly improves all the algorithms by providing adaptation to the specific scenario.

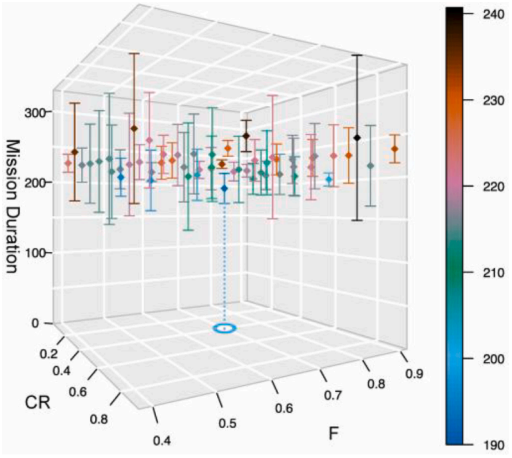

Fig. 11. DE/best/1/bin hyperparameters grid search, with the ACO-E algorithm and the Illegal Dump scenario.

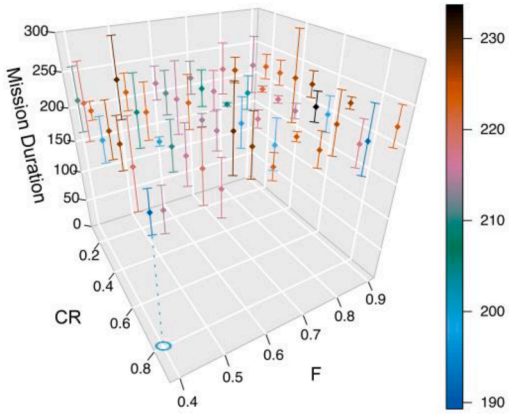

Fig. 12. DE/rand/1/bin hyperparameters grid search, with the ACO-E algorithm and the Illegal Dump scenario.

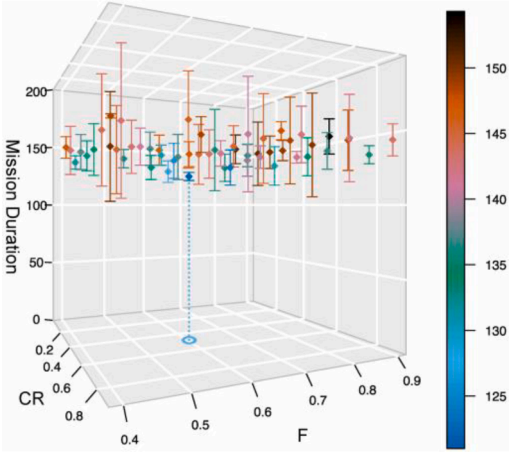

Fig. 13. DE/best/1/bin hyperparameters grid search, with the SFE-RR1 algorithm and the Illegal Dump scenario.

Figure 15 shows the average best fitness against number of generations of the optimization process.

Figure 16 and Figure 17 show the tracks left by drones coordinated by the SFE-RR1 before and after the DE. Here, a higher green intensity means more visits in the area. Clearly, the DE optimization leads to a more efficient exploration, focusing the search on the

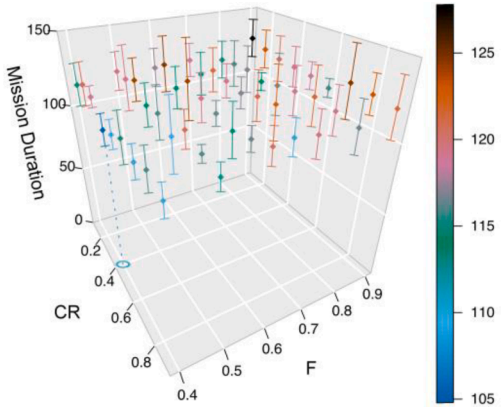

Fig. 14. DE/rand/1/bin hyperparameters grid search, with the SFE-RR1 algorithm and the Illegal Dump scenario.

Table 7
Swarm Exploration: mission duration before and after Differential Evolution

Scenario	Algorithm	Mission duration before DE	Mission duration after DE
Dump	SFE-RR1	185.97 ± 13.50	144.87 ± 09.62
"	ACO-E	317.10 ± 18.10	217.87 ± 09.56
Rural Mine	SFE-RR1	226.67 ± 51.03	159.53 ± 20.37
"	ACO-E	256.80 ± 14.08	205.00 ± 07.61
LPG Leak	SFE-RR1	191.57 ± 17.13	134.87 ± 05.09
"	ACO-E	215.43 ± 25.35	168.80 ± 04.04

Table 8
Swarm Exploration + Recruitment: mission duration, before and after Differential Evolution

Scenario	Algorithm	Mission duration before DE	Mission Duration after DE
Dump	SFE-RR3	251.87 ± 27.31	186.20 ± 04.02
"	FTS-RR3-E	331.23 ± 20.68	261.47 ± 09.10
"	PSO-RR3-E	396.00 ± 09.99	269.23 ± 03.55
"	ABC-RR3-E	575.87 ± 131.24	409.33 ± 19.93
Rural Mine	SFE-RR3	267.70 ± 24.51	193.90 ± 24.71
"	PSO-RR3-E	338.00 ± 61.86	236.67 ± 01.73
"	FTS-RR3-E	316.70 ± 30.45	262.00 ± 03.85
"	ABC-RR3-E	409.10 ± 26.56	318.00 ± 22.83
LPG Leak	SFE-RR3	220.10 ± 05.05	168.77 ± 07.44
"	PSO-RR3-E	459.77 ± 09.10	286.20 ± 21.12
"	FTS-RR3-E	482.43 ± 28.61	302.23 ± 14.45
"	ABC-RR3-E	832.43 ± 15.15	577.13 ± 19.20

regions of interest rather than on the initial positions.

To better show the improvements made by the optimization of the target discovery process, Figure 18 shows the average percentage of target found against time by the SFE, before and after the DE, over 10 trials.

Finally, to show the computational efficiency, let us consider the duration of DE for the different algorithms and for each scenario. The runtime of the DE depends linearly on the population size and on the number of generations. Let us fix the generations to 40 for all algorithms. Let us also consider that implementation is engineered for parallel computing. The hardware and software platforms used are: CPU Intel(R) Xeon(R) Gold 6140M at 2.2-2,3 GHz, Linux OS and Python (optimization) + Java/Netlogo (mission simulation). The optimization time of a mission depends on the scenario complexity and on the quality of the coordination mechanism, which are difficult to express. The optimization time can be empirically measured via the average DE runtime per scenario. Table 9 shows the average DE optimization time, over 3 runs, for 40 generations. The computational model of the SFE is the most efficient for both exploration and recruitment. In contrast, when considering the complexity in memory, a different situation appears. Table 10 shows the memory usage for each algorithm, for the Illegal Dump scenario. It is apparent that the SFE is much more expensive in terms of memory.

6. Conclusions

This paper focuses on the target search problem via swarms of robots, in complex or open environments. In this context, robots

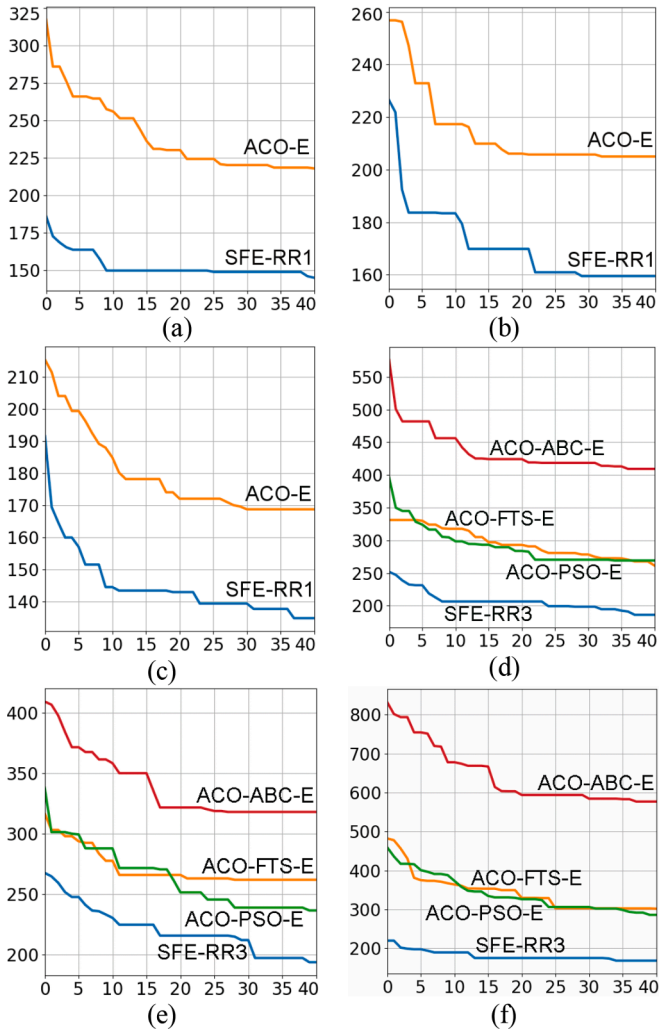

Fig. 15. Mission duration optimization: average best fitness against number of generations: exploration on (a) Dump, (b) Rural Mine, (c) LPG Leak, exploration + recruitment on (d), (e), (f)

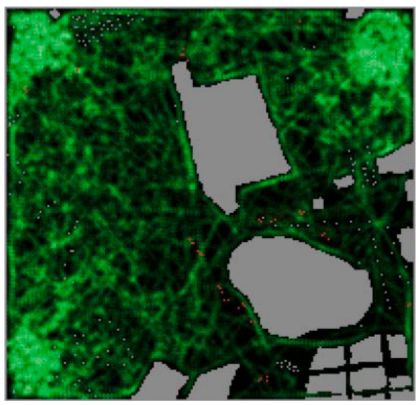

Fig. 16. Illegal Dump scenario: drones' trails before the DE

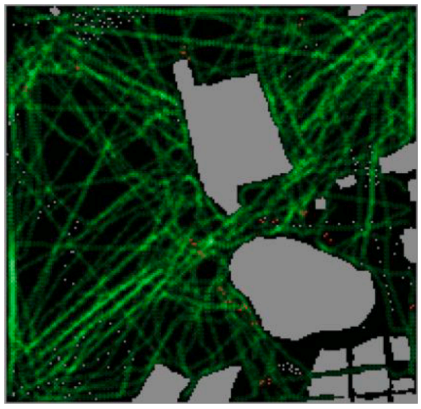

Fig. 17. Illegal Dump scenario: drones' trails after the DE

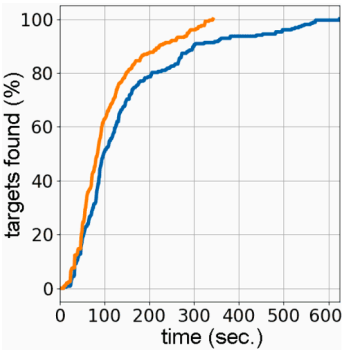

Fig. 18. Illegal Dump scenario: average percentage of targets found against time achieved by SFE, before (blue) and after (orange) DE

Table 9
Average DE optimization duration, for 40 generations

Scenario	Algorithm	Avg DE time	Pop. size
Dump	SFE-RR1	1h 04′ 01′′	56
"	ACO-E	9h 49′ 53′′	32
"	SFE RR3	1h 56′ 26′′	56
"	FTS-RR3-E	15h 29′ 39′′	44
"	PSO-RR3-E	1d 7h 29′ 41′′	44
"	ABC-RR3-E	19h 41′ 13′′	36
Rural Mine	SFE-RR1	1h 26′ 29′′	56
"	ACO-E	19h 34′ 52′′	32
"	SFE RR3	2h 7′ 22′′	56
"	FTS-RR3-E	14h 26′ 24′′	44
"	PSO-RR3-E	1d 19h 39′ 24′′	44
"	ABC-RR3-E	14h 42′ 13′′	36
LPG Leak	SFE-RR1	1h 4′ 16′′	56
"	ACO-E	1d 1h 49′ 55′′	32
"	SFE RR3	1h 59′ 12′′	56
"	FTS-RR3-E	22h 39′ 00′′	44
"	PSO-RR3-E	15h 16′ 18′′	44
"	ABC-RR3-E	1d 3h 13′ 45′′	36

cannot exploit static information on layout and targets locations, and therefore their coordination is fundamental for an efficient target discovery. To coordinate the swarm, the following popular bio-inspired swarm algorithms have been considered and made adaptive: ACO (inspired by ants) for exploration; FTS (fireflies), PSO (birds), and ABC (bees) for recruitment. Parametric adaptiveness is achieved via the DE optimization. The DE is able to find the best algorithmic parameters of a bio-inspired algorithm for improving the mission performance. In order to overcome the design constraints of bio-inspired approaches, an approach based on hyper-heuristics is also proposed. The proposed approach is called SFE because it is based on Stigmergy, Flocking and Evolution. Experimental results on real-world scenarios, carried out and released as a public testbed, show that the SFE significantly outperforms the adaptive bio-

Table 10
Memory usage at the end of the 1st DE generation

Algorithm	RAM (GB)	Pop. size	RAM (GB) per individual
SFE-RR1	317	56	5.66
ACO-E	127	32	3.97
SFE RR3	321	56	5.73
FTS-RR3-E	201	44	4.57
PSO-RR3-E	175	44	3.98
ABC-RR3-E	165	36	4.58

Algorithm 1
Differential Evolution algorithm

function differentialEvolution(*Robots, Obstacles, Targets*)
$t = 0$;
$P^{(0)}$ = initializePopulation();
for each genotype $\boldsymbol{p}_i^{(0)}$ **in** $P^{(0)}$ **do**
  $S_i^{(0)}$ = genotypeToSwarm($\boldsymbol{p}_i^{(0)}$);
  $f_i^{(0)}$ = simulateMission($S_i^{(0)}$, *Robots, Obstacles, Targets*);
do
  for each genotype $\boldsymbol{p}_i^{(t)}$ **in** $P^{(t)}$ **do**
    $\boldsymbol{m}$ = generateMutant($P^{(t)}$, $\boldsymbol{p}_i^{(t)}$);
    $\boldsymbol{p}_i^*$ = binomialCrossover($\boldsymbol{p}_i^{(t)}$, $\boldsymbol{m}$);
    S_i^* = genotypeToSwarm($\boldsymbol{p}_i^*$);
    f_i^* = simulateMission(S_i^*, *Robots, Obstacles, Targets*);
  for each genotype $\boldsymbol{p}_i^{(t)}$ **in** $P^{(t)}$ **do**
    if ($f_i^* < f_i^{(t)}$) **then**
      $\boldsymbol{p}_i^{(t+1)} = \boldsymbol{p}_i^*; f_i^{(t+1)} = f_i^*$;
    else
      $\boldsymbol{p}_i^{(t+1)} = \boldsymbol{p}_i^{(t)}; f_i^{(t+1)} = f_i^{(t)}$;
  $f_{min}^{(t+1)} = \min\{f_1^{(t+1)}, \ldots, f_N^{(t+1)}\}$;
  $t = t + 1$;
while (terminationCriterion($f_{min}^{(t)}$, t) = **false**);
return genotypeToSwarm($\boldsymbol{p}_{min}^{(t)}$);

Algorithm 2
Mutant vector generation for De/rand/1/bin

function generateMutant($P^{(t)}$, $\boldsymbol{p}_i^{(t)}$)
$\boldsymbol{p}'$ = randomExtraction($P^{(t)} \setminus \{\boldsymbol{p}_i^{(t)}\}$);
$\boldsymbol{p}''$ = randomExtraction($P^{(t)} \setminus \{\boldsymbol{p}_i^{(t)}, \boldsymbol{p}'\}$);
$\boldsymbol{p}'''$ = randomExtraction($P^{(t)} \setminus \{\boldsymbol{p}_i^{(t)}, \boldsymbol{p}', \boldsymbol{p}''\}$);
return $\boldsymbol{p}' + F \bullet (\boldsymbol{p}'' - \boldsymbol{p}''')$;

Algorithm 3
Binomial crossover

function binomialCrossover($\boldsymbol{p}_i^{(t)}$, $\boldsymbol{m}$)
k = randomInteger(1, K);
for each j-th gene $p_{j,i}^{(t)}$ **in** $\boldsymbol{p}_i^{(t)}$ **do**
  if (randomReal(0,1) < CR) or ($j = k$) **then**
    $w_j = m_j$;
  else
    $w_j = p_{j,i}^{(t)}$;
return $\boldsymbol{w}$;

heuristics, in both exploration and recruitment. The SFE is also faster in terms of optimization duration, although it requires more memory.

Authors' statement

Manuscript title: A hyper-heuristic methodology for coordinating swarms of robots in target search

All authors certify that they have participated sufficiently in the work to take public responsibility for the content, including participation in the concept, design, analysis, writing, or revision of the manuscript.

Mario G. C. A. Cimino, Domenico Minici, Manilo Monaco, Stefano Petrocchi and Gigliola Vaglini

Declaration of Competing Interest

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

Acknowledgments

Work partially supported by the Italian Ministry of Education and Research (MIUR) in the framework of the CrossLab project (Departments of Excellence).

References

[1] Dorigo M, Birattari M, Stutzle T. Ant colony optimization. IEEE Comput Intell Mag 2006;1(4):28–39.
[2] Alfeo AL, Cimino MGCA, De Francesco N, Lazzeri A, Lega M, Vaglini G. Swarm coordination of mini-UAVs for target search using imperfect sensors. Int Dec Tech 2018;12(2):149–62.
[3] Yang XS. Firefly algorithms for multimodal optimization. In: Proc. of 5th symposium on stochastic algorithms, foundations and applications (SAGA). Springer; 2009. p. 169–78.
[4] Karaboga D, Akay B. A comparative study of artificial bee colony algorithm. Appl Math Comput 2019;214(1):108–32.
[5] Senanayake M, Senthoorana I, Barca J, Chung H, Kamruzzamanc J, Murshedc M. Search and tracking algorithms for a swarm of robots: a survey. J Robot Auton Syst 2016;75:422–34.
[6] Cimino MGCA, Lazzeri A, Vaglini G. Improving the analysis of context-aware information via marker-based stigmergy and differential evolution. In: Proc. of Int. Conf. on Artificial Intelligence and Soft Computing (ICAISC); 2015. p. 341–52.
[7] Monaco M, (2021). Github platform, SFE repository, https://github.com/mlpi-unipi/sfe.
[8] Fujisawa R, Dobata S, Kubota D, Imamura H, Matsun F. Dependency by concentration of pheromone trail for multiple robots. In: Proc. of the 6th Int. Conf. on Ant Colony Optimization and Swarm Intelligence (ANTS). Springer; 2008. p. 283–90.
[9] Ducatelle F, Di Caro GA, Pinciroli C, Gambardella LM. Self-organized cooperation between robotic swarms. Swarm Intell 2011;5(2):73–96.
[10] Masár M, Zelenka J. Modification of PSO algorithm for the purpose of space exploration. In: Proc. of Int. Conf. on Intelligent Engineering Systems (INES). IEEE Press; 2012. p. 223–6.
[11] Contreras-Cruz MA, Ayala-Ramirez V, Hernandez-Belmonte UH. Mobile robot path planning using artificial bee colony and evolutionary programming. Appl Soft Comput 2015;30:319–28.
[12] Bayindir L. A review of swarm robotics tasks. Neurocomputing 2016;172:292–321.
[13] Hutter F, Hoos HH, Stützle T. Automatic algorithm configuration based on local search. In: Proc. of the Twenty-second AAAI Conference on Artificial Intelligence. AAAI Press; 2007. p. 1152–7.
[14] Eiben AE, Hinterding R, Michalewicz Z. Parameter control in evolutionary algorithms. IEEE Trans. Evol Comput 1999;3(2):124.
[15] Burke EK, Hart E, Kendall G, Newall J, Ross P, Schulenburg S. Hyperheuristics: An emerging direction in modern search technology. Handbook of Metaheuristics. Kluwer Press; 2003. p. 457–74.
[16] Burke EK, Gendreau M, Hyde M, Kendall G, Ochoa G, Özcan E, et al. Hyper-heuristics: a survey of the state of the art. J Oper Res Soc 2013;64(12):1695–724.
[17] Garrido P, Riff MC. Dvrp: A hard dynamic combinatorial optimisation problem tackled by an evolutionary hyper-heuristic. J Heuristics 2010;16(6):795–834.
[18] Koza JR, Poli R. Genetic programming. Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques. Kluwer 2005:127–64.
[19] Geiger CD, Uzsoy R, Aytug H. Rapid modeling and discovery of priority dispatching rules: An autonomous learning approach. J Sched 2006;9(1):7–34.
[20] Fukunaga AS. Automated discovery of local search heuristics for satisfiability testing. Evol Comput 2008;16(1):1–31. MIT Press.
[21] Ong YS, Lim MH, Zhu N, Wong KW. Classification of adaptive memetic algorithms: A comparative study. IEEE Trans Syst, Man, Cybernet, Part B 2006;36(1):141–52.
[22] Smith J. Adaptive and multilevel metaheuristics. chap Self-Adaptation in Evolutionary Algorithms for Combinatorial Optimisation. Springer; 2008. p. 31–57.
[23] Brest J, Greiner S, Boscovic B, Mernik M, Zumer V. Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems. IEEE Trans oEvol Comput 2006;10:646–57.
[24] Zaharie D. Influence of crossover on the behavior of differential evolution algorithms. Appl Soft Comput 2009;9:1126–38.
[25] Palmieri N, Yang XS, De Rango F, Maran S. Comparison of bio-inspired algorithms applied to the coordination of mobile robots considering the energy consumption. Neural Comput Appl 2019;31(1):263–86.
[26] Alfeo AL, Cimino MGCA, De Francesco N, Lega M, Vaglini G. Design and simulation of the emergent behavior of small drones swarming for distributed target localization. J Comput Sci 2018;29:19–33. 2018.
[27] Das S, Mullick SS, Suganthan PN. Recent advances in differential evolution – an updated survey. Swarm Evol Comput 2016;27:1–30.
[28] Cimino MGCA, Lega M, Monaco M, Vaglini G. Adaptive Exploration of a UAVs Swarm for Distributed Targets Detection and Tracking. In: Proc. of the 8th Int. Conf. on Pattern Recognition Applications and Methods (ICPRAM); 2019. p. 837–44.
[29] Alfeo AL, Cimino MGCA, Vaglini G. Enhancing biologically inspired swarm behavior: Metaheuristics to foster the optimization of UAVs coordination in target search. Comput Oper Res 2019;110:34–47.
[30] Pontiggia M, Landucci G, Busini V, Derudi M, Alba M, Scaioni M, et al. CFD model simulation of LPG dispersion in urban areas. Atmos Environ 2011;45(3913e3923).

Mario G. C. A. Cimino is an associate professor at the Department of Information Engineering of the University of Pisa (Italy). His research lies in the areas of information systems and artificial intelligence. He is (co-)author of about 70 scientific publications.

Domenico Minici is a PhD student in Smart Computing at the Department of Information Engineering of the University of Florence (Italy). He is also with the Department of Information Engineering of the University of Pisa (Italy). His research focuses on swarm intelligence and smart wearable systems for personalized healthcare.

Manilo Monaco is a Ph.D. student of the International Ph.D. Program in Smart Computing at the Department of Information Engineering of the University of Florence (Italy). He is also at the Department of Information Engineering of the University of Pisa (Italy). His research interests include Computational Intelligence, Swarm Intelligence, and their applications.

Stefano Petrocchi is a master's degree student in Artificial Intelligence and Data Engineering at the Department of Information Engineering of the University of Pisa (Italy). His interests include swarm robotics.

Gigliola Vaglini is full professor of Computer Engineering at "Dipartimento di Ingegneria della Informazione" of the University of Pisa. The main fields of her research activity are the specification and verification of concurrent and distributed systems, and the use of machine learning techniques for detecting malware in mobile systems and anomalies of industrial systems.