Multi-Objective Heuristics applied to Robot Task Planning for Inspection Plants

Itziar Landa-Torres*, Jesus L. Lobo*, Idoia Murua*, Diana Manjarres* and Javier Del Ser*,†,‡
*TECNALIA RESEARCH & INNOVATION, 48160 Zamudio, Bizkaia, Spain
Email: {itziar.landa, jesus.lopez, idoia.murua, diana.manjarres}@tecnalia.com
†University of the Basque Country UPV/EHU, Bilbao, Bizkaia, Spain
Email: {javier.delser}@ehu.eus
‡ Basque Center for Applied Mathematics (BCAM), Bilbao, Bizkaia, Spain

***Abstract*—Robotics are generally subject to stringent operational conditions that impose a high degree of criticality on the allocation of resources and the schedule of operations in mission planning. In this regard the so-called cost of a mission must be considered as an additional criterion when designing optimal task schedules within the mission at hand. Such a cost can be conceived as the impact of the mission on the robotic resources themselves, which range from the consumption of battery to other negative effects such as mechanic erosion. This manuscript focuses on this issue by presenting experimental results obtained over realistic scenarios of two heuristic solvers (MOHS and NSGA-II) aimed at efficiently scheduling tasks in robotic swarms that collaborate together to accomplish a mission. The heuristic techniques resort to a Random-Keys encoding strategy to represent the allocation of robots to tasks whereas the relative execution order of such tasks within the schedule of certain robots is computed based on the Traveling Salesman Problem (TSP). Experimental results in three different deployment scenarios reveal the goodness of the proposed technique based on the Multi-objective Harmony Search algorithm (MOHS) in terms of Hypervolume (HV) and Coverage Rate (CR) performance indicators.**

I. Introduction

Scheduling problems arise in many different fields of knowledge yielding from operational logistics to production processes. The main focus of scheduling aims at allocating jobs or tasks over time subject to mutually affecting constraints such as the maximum commit time beyond which all works should be completed or the limited availability of resources needed to finish the task, among others. During last decade an upsurge of new algorithmic techniques capable of efficiently dealing with scheduling problems of high dimensionality have arisen in different application domains, e.g. manufacturing [1], [2], [3] and service industry [4], [5], [6], [7], cloud computing [8] [9], [10], transport [11], [12], [13], [14], [15], [16].

Regarding the set of methods and techniques that have been proposed to deal with scheduling problems, meta-heuristics have been extensively applied as efficient solvers to seek for near-optimal solutions within less computation time than exact computation methods [17], [18], [19] and [20]. Indeed, the use of meta-heuristic algorithms enables the achievement of plausible solutions to complex optimization problems. In this context, bio-inspired solvers, which mimic self-learning behaviors observed in Nature when exploring solution spaces, have achieved a growing interest for solving job-shop scheduling problems, e.g Simulated Annealing (SA) [21], Particle Swarm Optimization (PSO) [22], Ant Colony Optimization (ACO) [23] and Genetic (GA) algorithms ([24], [25].

In addition, scheduling problems can be seen as optimal selection problems in which a subset of tasks or activities must be selected from a whole list of possible tasks. By the same way, it can also be formulated as a multi-objective optimization problem in which more than one objective – possibly conflicting with each other – has to be optimized. For instance, in robotics two possible conflicting objectives can be battery life versus commit time. Therefore, there exists no single solution that simultaneously optimizes both objective functions, but a set of Pareto-optimal solutions such that any slight improvement in one of the objectives involves a penalty in at least one of the remaining objectives. Due to the complexity of the underlying scheduling scenario, the generation of the Pareto optimal set can be computationally expensive and is not often possible. Thus, different meta-heuristic solvers that find their roots on Nature behavior have appeared in order to cope with multi-objective scheduling problems [26], [27]: EA's, GA [28], [29] and PSA [30] or more new ones as firefly [31] or bat algorithm [32]. Although they do not guarantee the identification of optimal solution sets, these meta-heuristic techniques attempt at obtaining a good approximation of Pareto-optimal sets.

This proposal is focused towards task scheduling in the case where a group of robots, along with the plant operator, collaborate with each other to accomplish a certain inspection and maintenance mission. In industrial missions there are several scenarios such as the one presented herein, where a collaboration among robots is required in order to accomplish the monitoring and supervision of industrial plants. In this proposal, the main idea is that, given a certain industrial plant comprised by multiple sensors and machinery that has to be motorized, measured and controlled, the algorithm has to determine the optimal set of vehicles to fulfill the mission, as well as calculate the path for each of the robots. Thus, the solution provided will comprise not only the robots, but also the paths to follow in order to accomplish the mission tasks optimally in terms of time (of measuring, displacements, among others) and cost (e.g. battery level). Nevertheless,

978-1-5090-4601-0/17/$31.00

restrictions such as the distance to the mission starting point, the status of the sensors on board, the weather conditions or even battery levels of the robots have to be taken into account when dealing with this multi-objective problem.

In this context, recent literature focused on multi-robot task scheduling problems can be found in [33]. Concretely, authors in [34] present a fuzzy scheduler capable of guiding robots dealing with real-time obstacle avoidance. Also related, in [35] a path planning and obstacle avoidance genetic-based algorithm for multiple robots is presented; a similar scenario tackled with Ant Colony algorithm is proposed in [36]. Similarly, in [37] an heuristic algorithm for solving a multi-robot task scheduling problem is addressed. In the same line of research, authors in [38] propose a two-phased multi-robot approach for task allocation and scheduling aimed at minimizing total execution time and its inoccupation (idle) time.

Within this field of research, this proposal takes a step beyond the state of the art by presenting the performance of different multi-objective algorithms for optimal collaborative task scheduling in a real time environment.

The mathematical formulation of this problem determines the optimality as a measure of two criteria: 1) the overall cost (accounting for different cost aspects of the schedule such as its impact on the energy consumption of the robots) should be minimized; 2) the minimization of total completion time of the mission is required. This optimization is accomplished while satisfying both mission and robot requirements as well as ensuring the total completion of the mission. Specifically, towards efficiently dealing with the aforementioned paradigm, this proposal exposes the practical performance of a multi-objective heuristic technique that utilizes a Random-Keys encoding to optimally represent the assignment of robots to tasks and build the final schedule. The heuristic scheduler has been designed to operate in a real scenario deployed in Steinkjer (Norway). Its application will highlight not only the practical feasible to real scenarios subject to cost and total time minimization criteria, but also the usability and advantages gained by utilizing it in real operation of the plant. That being so, the practical value of the proposed approach not only focuses on minimizing cost and distance, but also providing the operator an useful tool for task planning.

II. System Model and Problem Formulation

According to Figure 1 we consider a robotic swarm composed by N robots that are deployed over a geographical area in order to complete M inspection tasks at geographical coordinates $\{(x_m, y_m)\}_{m=1}^{M}$. These tasks are assumed not to be related to each other under any precedence constraint (i.e. tasks can be completed in any order along time). Each task imposes certain functionalities or skills on the robot scheduled for its completion. The overall set of functionalities available in the robotic swarm will be hereafter represented by $\mathcal{F} \doteq \{f_1, f_2, \ldots, f_F\}$, with $F \doteq |\mathcal{F}|$ denoting its cardinality. Following this notation, the subset of functionalities featured by robot $n \in \{1, \ldots, N\}$ will be denoted as $\mathcal{F}_R(n) \subseteq \mathcal{F}$, whereas the subset of skills required by task $m \in \{1, \ldots, M\}$ will be referred to as $\mathcal{F}_T(m) \subset \mathcal{F}$. It should be clear that a sine qua non condition for robot n to be scheduled for completing task n will be $\mathcal{F}_T(m) \subseteq \mathcal{F}_R(m)$, i.e. functionalities demanded by task n should be covered by the portfolio of functionalities of the robot at hand. Such skills are supposed to be set prior to the deployment of the robots and to be not transferable nor configurable during the mission.

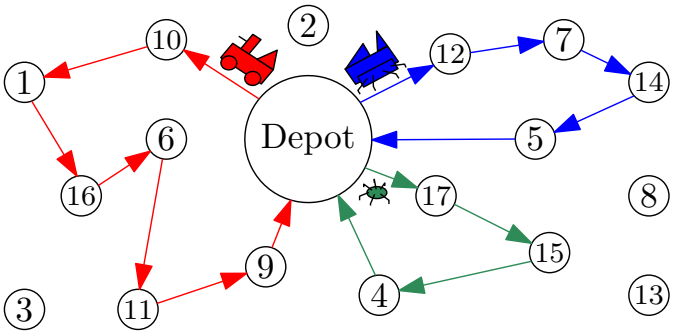

Fig. 1. Schematic diagram of a problem instance with $N = 3$ robots and $M = 17$ tasks. The plotted solution to the problem $\{\boldsymbol{\lambda}_{\blacksquare}, \boldsymbol{\lambda}_{\blacksquare}, \boldsymbol{\lambda}_{\blacksquare}\}$ – e.g. $\boldsymbol{\lambda}_{\blacksquare} = \{10, 1, 16, 6, 11, 9\}$ – is an ordered mapping of tasks to robots under a Pareto-optimal trade-off between cost of the mission and completion time.

Bearing in mind the above definitions, a mission schedule or plan consists of determining the allocation of robots to tasks not only in regards to which robot is enforced to accomplish each mission task, but also their sequencing along time for each robot. To jointly represent both variables a vector $\boldsymbol{\lambda}^n \doteq \{\lambda_1^n, \lambda_2^n, \ldots, \lambda_{M(n)}^n\}$ will represent the ordered sequence of the $M(n) \leq M$ tasks assigned to robot n, with $\lambda_m^n \in \{1, \ldots, M\}$ and $\lambda_m^n \neq \lambda_{m'}^{n'}$ if $m \neq m'$ and/or $n \neq n'$. Therefore, any given plan devised to complete the inspection mission can be represented by $\{\boldsymbol{\lambda}^n\}_{n=1}^{N}$, namely, the set of all schedules for the robotic swarm. This definition also accommodates the particular case when a given robot is left unscheduled due to e.g. an over-dimensioned number of robots for completing the tasks ($N \gg M$). This case would yield $M(n) = 2$ and $\boldsymbol{\lambda}^n = \{\lambda_0, \lambda_0\}$ for part of the robots selected under the criteria next defined. Finally, it is assumed that all robots depart from a single depot, assumption reflected in the above definitions by extending all schedules $\{\boldsymbol{\lambda}^n\}_{n=1}^{N}$ with an initial task λ_0 with coordinates (x_0, y_0) given by the position of the depot. Likewise, all deployed robots are commanded to return to the depot after completing all their scheduled tasks, resulting in $\boldsymbol{\lambda}^n = \{\lambda_0, \lambda_1^n, \ldots, \lambda_{M(n)}^n, \lambda_0\}\ \forall n \in \{1, \ldots, N\}$

To evaluate the quality of the mission plan two performance indicators over two different domains are defined. The first refers to the completion time of the mission, which will be given by the time at which all robots have finished their corresponding task schedules. Mathematically the time taken for robot n to complete task m, denoted as $T_m^{n,\checkmark}$ [seconds], will vary as a function of several factors (e.g. speed of the robot, processing power, battery consumption regime, etc) that depend roughly on the robot itself. Likewise, the completion time of the mission will also be subject to the sequence

of tasks and the time taken for robot m to move among the physical locations $\{(x_{\lambda_m^n}, y_{\lambda_m^n})\}_{m=1}^{M(n)}$ where its allocated tasks are to be developed. In this regard we will refer as $T^n(m \rightsquigarrow m')$ [seconds] to the time taken for robot n to travel from coordinate (x_m, y_m) to $(x_{m'}, y_{m'})$ subject to its speed of travel, trajectory and ability to cope with obstacles along its path (e.g. steep areas, pebbly areas and other obstructions alike). Based on this notation the overall completion time of mission $\{\boldsymbol{\lambda}^n\}_{n=1}^N$ will be given by

$$T_{\circlearrowleft}(\{\boldsymbol{\lambda}^n\}_{n=1}^N) = \max_n \sum_{m=1}^{M(n)+1} \left(T^n(\lambda_{m-1}^n \rightsquigarrow \lambda_m^n) + T_{\lambda_m^n}^{n,\checkmark} \right), \quad (1)$$

such that $T_m^{n,\checkmark} = 0$ for $m = \lambda_{M(n)+1}^n$ as once returned to the depot robots are not commanded to perform any task.

The second performance indicator relates to the cost associated to the mission plan, which relates to the cost associated to the inspection task and the use of robot n in the swarm. The former cost concept is set by 1) the radio coverage level R_m [%] of the geographical point over which the inspection task is to be performed, which impacts on the available transmission bandwidth for transmitting the information monitored on site via camera, sensors and other sensors installed on board; and 2) the nature of data collected during the task at hand, i.e. if mission m requires video to be transmitted to the control center the required bandwidth will be higher than that of sending low-rate gas measures. All in all, the combination of the required (task) and available (location) bandwidth entails costs whenever the information to be captured demands high-rate wireless equipment to be installed on the robot and/or the radio coverage level of the area is low enough to further justify this additional equipment. Second, further costs arise when scheduling different robots due to e.g. varying battery levels, even between any pair featuring the same subset of functionalities for the mission. Such costs associated to the mission will be hence given by

$$C_{\circlearrowleft}(\{\boldsymbol{\lambda}^n\}_{n=1}^N) = \sum_{\substack{n=1 \\ M(n)>2}}^{N} \left(C_R(n) + \sum_{m=1}^{M(n)} C_T(\lambda_m^n) \right), \quad (2)$$

where $C_R(n)$ and $C_T(m)$ respectively denote the cost for robot n and task m. The latter cost is computed as $C_T(m) \doteq C_{BW}(m) + L(m)$, where $C_{BW}(m)$ denotes the cost associated to installing a radio interface fulfilling the maximum rate required to transmit the information captured during task m; and $L(m)$ denotes the relative coverage level available at location (x_m, y_m) with respect to an ideal communication channel exclusively undergoing distance-dependent free space losses. Values of all cost concepts are normalized to the range $[0, 100]$ so as to achieve a properly balanced cost definition.

Before formally posing the optimization problem tackled in this manuscript it must be noted that the above definitions implicitly assume that all inspection tasks must be addressed and completed by the robot swarm, i.e. $\cup_{n=1}^N \boldsymbol{\lambda}^n = \{1, \ldots, M\}$. This may not be feasible when the number of available robots N is significantly lower than M or the number of qualified robots in the swarm is not high enough to cover all the skill requirements of the mission commit. As we will later tackle these two hypothesis and propose different algorithmic approaches to deal with them, we will relax this requirement in the problem statement. Therefore, the multi-objective problem can be formulated as the discovery of the Pareto optimal mission plans $\{\{\boldsymbol{\lambda}^{n,*}\}_{n=1}^N$ such that $\boldsymbol{\lambda}^{n,*} = \{\lambda_0, \lambda_1^{n,*}, \ldots, \lambda_{M(n,*)}^{n,k}, \lambda_0\}$ that differently balances between mission time $T_{\circlearrowleft}(\cdot)$ and cost $C_{\circlearrowleft}(\cdot)$, i.e.

$$\{\boldsymbol{\lambda}^{n,*}\}_{n=1}^N = \min_{\{\boldsymbol{\lambda}^n\}_{n=1}^N} \left[T_{\circlearrowleft}(\{\boldsymbol{\lambda}^n\}_{n=1}^N),\ C_{\circlearrowleft}(\{\boldsymbol{\lambda}^n\}_{n=1}^N) \right], \quad (3)$$

$$\text{subject to } \lambda_m^n \neq \lambda_{m'}^{n'} \text{ if } m \neq m' \text{ or } n \neq n', \quad (4)$$

where $\min_{\mathbf{x}}[f_1(\mathbf{x}), f_2(\mathbf{x})]$ represents that the optimization criterion is driven by the Pareto trade-off between conflicting objectives $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ set by different values of the optimization variable $\mathbf{x}$. The solution to the above problem is a set of solutions that trades one objective for another differently yet optimally in terms of Pareto dominance.

A. Multi-objective Harmony Search Algorithm (MOHS)

The Harmony Search (HS) optimization algorithm was first proposed by Geem et al. in [40] and has been lately applied to diverse applications and problems that range from multiple disciplines, such as Energy [41], Games [42], vehicle routing [43], [44] and Health operations [45], among others.

This paper presents a multi-objective version of the HS algorithm that simultaneously minimizes the total time and cost of a certain mission. As HS is a population-based algorithm its parameters (HMCR, PAR and RSR) are applied to a set of possible solutions $\{\mathbf{H}(k)\}_{k=1}^K$ (denoted as Harmony Memory), which aim at iteratively modifying such solutions towards regions of higher optimality. Following the naive notation of HS, a possible candidate set $\mathbf{H}(k)$ is denoted as *harmony*, whereas *note* stands for any of its N entries.

The Random-Keys (RK) encoding [46] represents each note of the harmony $H_{k,n}$ (with $n \in \{1, \ldots, N\}$ and $k \in \{1, \ldots, K\}$) as a real positive number; the integer part $\lfloor H_{k,n} \rfloor$ refers to the index of the robot that will accomplish task $\texttt{TASK}_n$, whereas the fractional part $H_{k,n} - \lfloor H_{k,n} \rfloor$ establishes the order of the tasks in such a way that tasks with lower fractional part are executed earlier than those with higher fractional part. In this optimization problem each note is encoded based on a simplified version of this encoding, as the integer part, responsible for ordering the tasks, is combined with a genetic TSP (Travelling Salesman Problem) solver in order to fasten the entire process. The reason for utilizing the TSP solver is that, at this point, the best order for the inspection tasks is the optimal path between them (by terms of minimizing distances), so the TSP will lead to optimal and solutions.

To decode the RK-encoded individual all notes sharing the same value for their integer part are grouped and sorted in increasing order of the value of their fractional part. This process results in the planning of tasks for every robot.

For instance, the solution vector $\mathbf{H}(k) \triangleq \{H_{k,n}\}_{n=1}^{4} = \{2.35,\ 1.96,\ 2.73,\ 1.14\}$ corresponds to the schedule:

Robot 1: $\texttt{TASK}_4$, $\texttt{TASK}_2$
Robot 2: $\texttt{TASK}_1$, $\texttt{TASK}_3$

Following this encoding, the integer part makes the task-robot assignment and once the tasks for all the robots are identified, the optimal route for each robotic is calculated separately. Thus, the fractional part is not included on this approach.

In the literature RK encoding has been utilized to represent solutions of evolutionary solvers that handle task plans, often improved further by means of local search procedures [47]. The main benefit of using this representation method based on trees is that it can be utilized to generate initial feasible individuals that remain feasible upon crossover and mutation and as such, do not require any repairing operator to ensure feasibility [48] along the iterative process. That being so, the authors in [49] propose a GA based approach for topology optimization. Other recent contributions have also utilized RK for solving the job-shop problem; in [28] a computer simulation of a plant that provides a quantitative measure of the optimality fitness that guides the search process is presented. In a similar approach, HS has been previously combined with RK in [50], but applied to a single objective optimization scheduling problem for solving the aforementioned job-shop paradigm.

1) General Scheme: The steps of the proposed multi-objective HS algorithm are the following:

A. The *initialization* step is only performed at the first iteration. At this point, the Harmony Memory $\mathbf{H}(k)$ is randomly filled: the integer part, which determines the robot that executes each task n with $n \in \{1, \ldots, N\}$, is taken uniformly at random from $\{1, \ldots, M\}$.
B. In the *improvisation* procedure, two probabilistic operators described above are sequentially applied to each note so as to produce a new set of K improvised harmonies, as it is thoughtfully detailed in Section II-A1. The optimal routes for each robot are calculated with a genetic based TSP algorithm and the harmonies are updated with the reordering of the tasks.
C. In the *evaluation* step, the candidate solutions are evaluated under both metrics (total completion time and the cost) according to the Expressions (1) and (2).
D. Once all the new solutions are evaluated and its metric computed, each solution is assigned a rank and a crowding distance value. The solutions with less rank value and largest crowding distance will be selected in order to remain in the HM along the iterative process. As explained in [39] less rank value yields to solutions with optimized metrics, whereas largest crowding distance aims at obtaining larger Pareto fronts.
E. This process is repeated until a *stop criterion* is reached. In this proposal, the algorithm iterates by returning to step B until the maximum number of iterations $\mathcal{I}$ is reached. When the algorithm stops, the set of candidate solutions stored in the HM, which comprise the final approximation of the Pareto front, corresponds to the solutions for the scheduling problem presented in this manuscript.

2) Improvisation parameters: The improvisation procedure that generates new candidate solutions along the iterative process is comprised by two operators that are sequentially applied to each note of the harmony, namely:

- The *Harmony Memory Considering Rate* (HMCR $\in [0, 1]$) operator applied to a certain note establishes that the new value of such note is taken randomly from the values of such note (same position in the solution vector) in the rest of harmonies of the HM. Thus, under the HMCR probability, the new value for a certain note is uniformly selected at random from the discrete set $\{1, \ldots, M\}$, where M stands for the total number of robots. As the HMCR operator is only applied to the integer part of the note, changes arising from its application will only affect to the robot-task assignment.
- The *Random Selection Rate* (RSR $\in [0, 1]$) represents the probability that the new value for a certain note is generated by making a subtle modification of its previous value. This operator is applied also to the integer part of the note so that it aims at changing randomly the assignment of a task to a certain robot. This parameter can include in the solution vectors robots that were not included before, so it prevents from early convergence of the algorithm.

III. Developed tool

The aim of the tool presented in this paper is to handle several robots and plan their actions so as to optimize the whole mission cost and time using a tool that helps the operator making suitable plans. Thus, the focus is put on the "plant monitoring" scenario, in which the mission comprises: 1) inspecting the plant, 2) taking measurements, 3) navigating around and 4) providing information to the operator. The exchanged information includes diverse actions such as the acquisition of images and/or videos and the delivery of measurements from different sensors onboard. That being so, to map the real scenario and provide real-time information to the operator, apart from defining the tasks that comprise the mission, the first step is to depict the set of available robots in the exact area to be inspected, as shown in Figure 2. The developed tool makes possible to select a subset of robots to complete the mission and provides real information of its current positions. Additionally, weather forecast and relevant information from previous inspections is provided before tracing the new inspection plan. Once, the robots are located, the operator has the opportunity to discard in advance some tasks or robots for the mission. Examples of tasks involved in the monitoring of an area are "move to waypoint", "follow row", "measure" or "take samples" (e.g. from the gas fuel), "acquire stereo vision data" (images and/or video), "send communication", and other duties alike. It is important to remark that differences among robots are related not only to their capabilities to perform the tasks (i.e. not all robots can accomplish any task), but also to their price and incurred

cost when undertaking a certain task; each robot may require different time and battery cost to carry out a certain task.

Once the mission is selected, the weather alerts, the past conditions and the task and robots supervised by the operator, the algorithm is launched on real time operation. Then, the output provided by the scheduler represents a set of solutions in which each of them corresponds to a task plan for each robot in the mission, which includes their task sequence and planned trajectory. The computed plans are shown in a mission management graphical user interface. For this setup, all those tasks are addressed with a Graphical User Interface (GUI) that show the mission input and output on a geographical information system, along with a Gantt chart of the task schedule for each robot, as depicted in Figure 2.

Thus, making use of the GUI, the operator is provided with a set of solutions and information related to each of them (time, cost, inspection points involved, a map of the route in google maps, among others), in order to facilitate the selection of the optimal solution. As explained before, each solution the algorithm provides has optimized the task schedule for each member of the entire robotic swarm and the operator selects the best candidate among the proposed solutions. The output of the solver, now integrated into the human-machine interface, has revealed that the algorithm is capable of planning the mission of up to four robots, covering the entire area to be inspected with computation time in the order of a few minutes' time. In summary, the GUI entails the following steps:

- Information retrieval: the operator gets real time information related to previous inspections, weather alerts and robot location.
- Mission definition: the system informs the operator which robots are available and their configurations, as well as the inspection points presumed to be involved in such mission. The operator can discard from the display in the GUI the robots and points set that wants to remove from the mission.
- Multi-objective scheduling algorithm: to optimize the whole mission and coordinate the robots, the system needs a planning model describing the available objects and their possible tasks. Given this model, the proposed heuristic algorithms compute the best sequence of actions performed by the robots, i.e. the plan.
- Gantt chart view of the task plans for each robot: the mission plan consists of a list of tasks assigned to each robot. Plans are shown as paths on the map and Gantt charts, showing the duration of the tasks, the order of execution for each robot and depicting the route in the map.
- Taking into account the information provided, the operator selects a plan from the GUI and stores it in the database; the plan can be executed on real time or queued in the system.

This study is focused not only on the multi-objective scheduling algorithm themselves, but also in real time operation with robots. The results provided in the next subsection describe quantitatively the task schedules produced by the algorithm, as well as results obtained in the real environment in Steinkjer (Norway).

Fig. 2. Real simulation setup deployed in Steinkjer (Norway).

IV. Experimental Results

In order to assess the performance rendered by the multi-objective MOHS approach proposed in this paper, a comparison study towards the well-known Non-dominated Sorting Genetic Algorithm (NSGA-II [39]) in a real scenario in Steinkjer (Norway) will be presented and discussed. In this real scenario a total of $N = 4$ robots are employed for accomplishing a specific mission composed of different tasks ($M = 200$) linked to inspection points. As stated in Section III each robot is capable of executing certain tasks in a specific inspection point based on its capacity and properties. As robots

have distinct functionalities and usages, a different cost and time is associated per pair (m, n), i.e. robots with higher cost per task require less time to execute the task, as well as different time to move from a certain inspection point to another. However, there are tasks that can be performed by different robots, and the selection of one or another depends on the total list of tasks and the availability of each robot.

Simulation results consider: 1) a baseline scenario in which robots are located relatively close to each other and without battery limitations; 2) a distance-based scenario in which robot $n = 1$ (the fastest and most expensive one) is located far from the mission area; and 3) a battery-limited scenario in which robot $n = 4$ (the cheapest but slowest one) undergoes a battery capacity restriction. Both multi-objective approaches are configured with the same number of Monte Carlo simulations, i.e. 10 in all cases, and maintain a memory of 50 candidate solutions. This ensures fairness in the comparison between such approaches as the number of fitness evaluations is the same among solvers. The values of the operators for all approaches have been optimized in order to obtain the best performance in the baseline scenario, and are extended to the remaining use cases (battery-limited and distance-based scenarios). Regarding MOHS, the values of the HMCR and PAR operators are set to 0.3 and 0.1, respectively. NSGA-II employs a uniform crossover of 0.3 and Gaussian mutation with probability of 0.1. Both algorithms run over 20 Montecarlos.

There are different multi-objective performance metrics that can be employed in order to evaluate the quality of the approximated Pareto fronts obtained by multi-objective approaches. On the one hand, two Pareto fronts can be compared in terms of diversity metrics. In this regard, Table I shows the hypervolume (HV) and the normalized hypervolume (HV norm) metric (%) with a common reference point per real case study. It is widely known that distribution and spread in multi-objective techniques are a highly sought characteristic: distribution refers to the relative distance among solutions, whereas spread stands for the range of values covered by the estimated Pareto front. In this regard, the HV metric, which calculates the fraction of space covered by solutions in the objective space with respect to a cuboid given by reference points, blends both aspects together into a single numerical score. The results obtained in all of the scenarios reveal a higher HV value when employing the MOHS approach as opposed the NSGA-II. That being so, MOHS has a better explorative behavior that permits to explore a wider range of solutions with different number of robots, i.e. solutions with similar cost metric values as per Expression (2) but that require slightly more time – corr. (1) – to accomplish the same mission. NSGA-II offers less diversity of solutions than MOHS and renders a worse performance than MOHS in terms of the HV metric in all scenarios.

On the other hand, cardinality metrics refers to the number of solutions that exists in the resultant Pareto Front; intuitively, a high number of solutions – and hence a high value of such metrics – is preferred. In this context, Table II presents the number of non-dominated solutions (NP) in the resulting Pareto Fronts per multi-objective approach and use case. As can be shown, MOHS obtains higher number of non-dominated solutions in all scenarios. This is due to the explorative capability of MOHS, which allows exploring a wide range of distinct solutions in the search space. As a result, this solver obtains more diversity of results. However, it is adverted that the gain of MOHS with respect to NSGA-II is reduced in Scenario 2, where NSGA-II and MOHS attain similar values; the reason behind this is because Scenario 2 can just operate with 3 robots (one of them is far located) and the explorative capability of the algorithm is less required for finding neal-optimal solutions.

TABLE I
HYPERVOLUME AND NORMALIZED HYPERVOLUME (HVNOR.) (%) WITH A COMMON REFERENCE POINT PER MULTI-OBJECTIVE APPROACH AND REAL USE CASE SCENARIO (1: BASELINE SCENARIO; 2: DISTANCE-BASED SCENARIO AND 3) BATTERY-LIMITED SCENARIO).

	MOHS HV	NSGAII HV	MOHS HVnor.	NSGAII HVnor.
1	2723784736	2070765328	0.008463	0.006232
2	3409253248	1113950776	0.011139	0.004611
3	2180496742	2154872832	0.007907	0.006541

TABLE II
COVERAGE RATE (CR) (%), NUMBER OF NON-DOMINANT POINTS (NP), AND TOTAL POINTS IN THE PARETO FRONT (PP) PER MULTI-OBJECTIVE APPROACH AND REAL USE CASE SCENARIO (1: BASELINE SCENARIO; 2: DISTANCE-BASED SCENARIO AND 3) BATTERY-LIMITED SCENARIO).

	MOHS CR	NSGAII CR	MOHS NP	NSGAII NP	MOHS PP	NSGAII PP
1	33	10	13	4	24	15
2	25	21	7	6	15	13
3	26	3	8	1	17	17

Finally, the Coverage Rate metric (CR) (%) also presented in Table II, reflects the number of solutions within each Pareto Front that are non-dominated by any solution in the rest of fronts. As can be shown MOHS is capable of obtaining the highest percentage of non-dominated solutions in all scenarios due to its capability to extensively explore the search space. In this case, the same effect is adverted when analyzing the number of points in scenario 2: results show that the values provided by both algorithms are similar because of the search space reduction due to robot restrictions in such scenario.

Making a deeper analysis of all the results obtained, it can be adverted in Figure IV that in Scenario 1, as it has not time or battery restrictions, the plans are executed by four robots in parallel. This is the optimal approach as the routes of the robots are independent, so the overall time of the mission is calculated taken into account the last robot to come back to the hut. Intuitively, the first scenario utilizes the four robots in parallel, attaining 15 points in the Pareto frontier with the GA and 24 the MOHS, as it can be seen in Table II. This results further evidence that the HS is more explorative and finds more diverse solutions seeking on the search space of the problem, as the HV, HV norm, CR, NP and PP parameters indicate. In this scenario the difference between MOHS and NSGA-II is

clear.

Nevertheless, regarding the results obtained in Scenario 2, in which $n = 1$ (the fastest and most expensive one) is located far away, both algorithms calculate the plan just with three robots, and hence the search space is reduced with respect to the baseline scenario. Accordingly, less solutions are found (13 the NSGA-II and 15 MOHS); one can see that MOHS still follows a more explorative behavior, but forfeits the advantage attained in the first scenario.

The last Scenario shows the case in which robot $n = 4$ (the slowest and less expensive one) is running out of battery. That being so, this robot can be included in the mission, but just for few tasks. In this case again, the search space allows the robots to operate, but the number of task assigned to $n = 4$ is reduced; this problem is more restricted than the baseline presented in Scenario 1, but less than Scenario 2. In this case, both algorithms find 17 points in the Pareto optimal, evidencing that this case has middle complexity and the explorative behavior of the MOHS does not offer clear advantages over the NSGA-II.

More in detail, we can see that the differences between scenarios mentioned previously can be adverted in the Pareto fronts: it can be seen that the results obtained in the first scenario are the ones that achieve faster solutions (further minimize time axis x), as all robots can operate fully in parallel. On the contrary, one can see that the results obtained with both algorithms for Scenario 2 are the ones that less minimize time axis; the plans calculated for this scenario take the longest as just comprise three robots. Consequently, the costs (y axis) are a bit reduced with respect to Scenario 1 because the robot left outside of the plan is the most expensive one and hence, the task have been assigned just to 3 robots, but less costly. At last, Scenario 3 can utilize the four robots, but taking into account that one of them is not fully loaded. This robot $n = 4$ is slow but cheap, and henceforth, using it for few tasks will incur not only into higher costs in the overall mission, but also in higher overall time. Despite the four robots operate in parallel, if one is running out of battery, the rest will have to assume more tasks than in the baseline Scenario 1, and will take longer to accomplish the mission.

V. Concluding Remarks

This work presents a task assignment and scheduling problem performed collaboratively by robot swarms. This paradigm is envisioned within monitoring and inspection missions located in a plant. The main objective herein aims at minimizing two conflicting criteria: the overall time of the mission and its cost. Regarding the cost of a certain mission, it can be defined as a numerical score related to the impact of the assignment of a task to a certain robot entails (such as e.g. battery consumption). In order to efficiently deal with this multi-objective paradigm, two different multi-objective metaheuristics, namely NSGA-II and MOHS, have been designed both using a RK encoding strategy. This way, the solutions simultaneously represent both the mapping from tasks to robots and the scheduling of tasks within every robot commit.

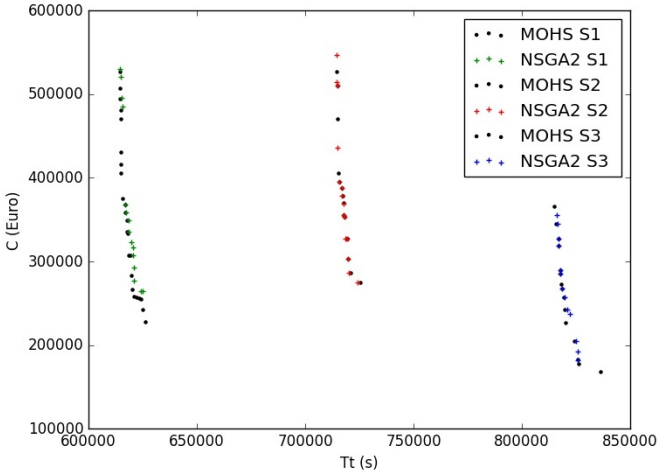

Fig. 3. Pareto fronts obtained for the three scenarios.

The performance of the multi-objective solvers has been assessed over three real-based scenarios deployed in Steinkjer (Norway) in terms of different multi-objective performance indicators, which quantify the cardinality, distribution and spread of the obtained non-dominated solutions. In the proposed paradigm, the importance of achieving a wide Pareto front and diversity of results is a key point. In this regard, MOHS attains a higher explorative behavior than NSGA-II rendering the best performance metrics in terms or Hypervolume and Coverage Rate in all scenarios, especially in those under operational constraints as it explores the search space efficiently and consequently yields the best approximation of the optimal Pareto frontier.

Future research will be focused on creating a full decision support system (DDS) in order to further help the operator in the plant maintenance. First steps towards constructing a full DDS will work towards storing information about past missions and utilize such acquired knowledge in order to advise the operator when the Pareto front is calculated and just one solution has to be chosen. Furthermore, in order to attain better plans, further constraints will be included in the optimization problem, such as the inclusion of relationships of dependence between tasks, the availability of charging depots in the mission area or the transfer of unfinished tasks between robots. All these restrictions and improvements will be included in the problem statement in the near future and tested in the real scenario located in Steinkjer.

Acknowledgments

This work has been supported by ECSEL in the frame of the European FP7 research project Reconfigurable ROS-based Resilient Reasoning Robotic Cooperating Systems (R5COP), grant agreement no. 621447 / 2014.

References

[1] Godinho F., M., Barco, C. F., Neto, R. F. T. (2014). Using Genetic Algorithms to solve scheduling problems on flexible manufacturing

systems (FMS): a literature survey, classification and analysis. Flexible Services and Manufacturing Journal, 26(3), 408-431.

[2] Priore P., Fuente D., Puente J., Parreo J. (2006) A comparison of machine-learning algorithms for dynamic scheduling of flexible manufacturing systems. Engineering Applications of Artificial Intelligence, 19(3), 247-255.

[3] Ouelhadj, D., Petrovic, S. (2009). A survey of dynamic scheduling in manufacturing systems. Journal of scheduling, 12(4), 417-431.

[4] Bertel, S., Billaut, J. C. (2004). A genetic algorithm for an industrial multiprocessor flow shop scheduling problem with recirculation. European Journal of Operational Research, 159(3), 651-662.

[5] Kallrath, J. (2002). Planning and scheduling in the process industry. OR spectrum, 24(3), 219-250.

[6] Amaro, A. C. S., Barbosa-Pvoa, A. P. F. (2008). Planning and scheduling of industrial supply chains with reverse flows: A real pharmaceutical case study. Computers & Chemical Engineering, 32(11), 2606-2625.

[7] Moon, C., Lee, Y. H., Jeong, C. S., Yun, Y. (2008). Integrated process planning and scheduling in a supply chain. Computers & Industrial Engineering, 54(4), 1048-1061.

[8] Li, J., F., Peng J. (2011). Task scheduling algorithm based on improved genetic algorithm in cloud computing environment. Jisuanji Yingyong Journal of Computer Applications, 31(1), 184-186, 2011.

[9] Shen, L. J., Liu, L., Lu, R., Chen, Y. T., Tian, P. P. (2012). Task Scheduling in Cloud Computing Based on Improved Immune Evolutionary Algorithm. Computer Engineering, 9, 62.

[10] Gu, J., Hu, J., Zhao, T., Sun, G. (2012). A new resource scheduling strategy based on genetic algorithm in cloud computing environment. Journal of Computers, 7(1), 42-52.

[11] Nuortio, T., Kytjoki, J., Niska, H., Brysy, O. (2006). Improved route planning and scheduling of waste collection and transport. Expert systems with applications, 30(2), 223-232.

[12] Chakroborty, P., Deb, K., Subrahmanyam, P. S. (1995). Optimal scheduling of urban transit systems using genetic algorithms. Journal of transportation Engineering, 121(6), 544-553.

[13] Christiansen, M., Fagerholt, K., Ronen, D. (2004). Ship routing and scheduling: Status and perspectives. Transportation science, 38(1), 1-18.

[14] Slater, A. (2002). Specification for a dynamic vehicle routing and scheduling system. International Journal of Transport Management, 1(1), 29-40.

[15] Teodorovic, D., Radivojevic, G. (2000). A fuzzy logic approach to dynamic dial-a-ride problem. Fuzzy sets and systems, 116(1), 23-33.

[16] Wren, A., Wren, D. O. (1995). A genetic algorithm for public transport driver scheduling. Computers & Operations Research, 22(1), 101-110.

[17] Jin, Z., Yang, Z., Ito, T. (2006). Metaheuristic algorithms for the multistage hybrid flowshop scheduling problem. International Journal of Production Economics, 100(2), 322-334.

[18] Kolisch, R., Hartmann, S. (1999). Heuristic algorithms for the resource-constrained project scheduling problem: Classification and computational analysis. Springer US, 147-178.

[19] Hartmann, S., Kolisch, R. (2000). Experimental evaluation of state-of-the-art heuristics for the resource-constrained project scheduling problem. European Journal of Operational Research, 127(2), 394-407.

[20] Back T., Hammel U., Schwefel H. P. (1997). Evolutionary computation: Comments on the history and current state. IEEE transactions on Evolutionary Computation, 1(1), 3-17.

[21] Low, C. (2005). Simulated annealing heuristic for flow shop scheduling problems with unrelated parallel machines. Computers & Operations Research, 32(8), 2013-2025.

[22] Liu, B., Wang, L., Jin, Y. H. (2007). An effective PSO-based memetic algorithm for flow shop scheduling. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 37(1), 18-27.

[23] Rajendran, C., Ziegler, H. (2004). Ant-colony algorithms for permutation flowshop scheduling to minimize makespan/total flowtime of jobs. European Journal of Operational Research, 155(2), 426-438.

[24] Cheng R., Gen M., Tsujimura Y. (1996). A tutorial survey of job-shop scheduling problems using genetic algorithms. Representation. Computers & industrial engineering, 30(4), 983-997.

[25] Pezzella, F., Morganti, G., Ciaschetti, G. (2008). A genetic algorithm for the flexible job-shop scheduling problem. Computers & Operations Research, 35(10), 3202-3212.

[26] Deb, K., Sindhya, K., Hakanen, J. (2016). Multi-objective optimization. In Decision Sciences: Theory and Practice. CRC Press, 145-184.

[27] Sun Y., Zhang C., Gao L., Wang X. (2011). Multi-objective optimization algorithms for flow shop scheduling problem: a review and prospects. The International Journal of Advanced Manufacturing Technology, 55(5), 723-739.

[28] Deb, K., Pratap, A. Agarwal, S. Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE transactions on evolutionary computation, 6(2), 182-197.

[29] Long J., Zheng Z., Gao, X. Pardalos, P. (2016). A hybrid multi-objective evolutionary algorithm based on NSGA-II for practical scheduling with release times in steel plants. Journal of the Operational Research Society, 67(9), 0.

[30] Coello, C. A. C., Pulido, G. T., Lechuga, M. S. (2004). Handling multiple objectives with particle swarm optimization. IEEE Transactions on evolutionary computation, 8(3), 256-279.

[31] Yang, X. S. (2013). Multiobjective firefly algorithm for continuous optimization. Engineering with Computers, 29(2), 175-184.

[32] Yang, X. S. (2011). Bat algorithm for multi-objective optimisation. International Journal of Bio-Inspired Computation, 3(5), 267-274.

[33] Arai, T., Pagello, E., Parker, L. E. (2002). Editorial: Advances in multi-robot systems. IEEE Transactions on robotics and automation, 18(5), 655-661.

[34] Kasim M. Al-Aubidy, Mohammed M. Ali and Ahmad M. Derbas. (2015). Multi-robot task scheduling and routing using neuro-fuzzy control. 12th International Multi-Conference on Systems, Signals & Devices, 1-6.

[35] Pires, E. S., Machado, J. T., Moura Oliveira, P. B. (2004). Robot trajectory planning using multi-objective genetic algorithm optimization. In Genetic and Evolutionary Computation Conference. Springer Berlin Heidelberg, 615-626.

[36] Liu, S., Mao, L., Yu, J. (2006). Path planning based on ant colony algorithm and distributed local navigation for multi-robot systems. In Mechatronics and Automation, Proceedings of the 2006 IEEE International Conference, 1733-1738.

[37] Zhang Y., Parker L.E. (2013). Multi-robot task scheduling. 2013 IEEE International Conference on Robotics and Automation, 2992-2998.

[38] Maoudj A., Bouzouia B., Hentout A., Toumi R. (2015). Multi-agent approach for task allocation and scheduling in cooperative heterogeneous multi-robot team: Simulation results. IEEE 13th International Conference on Industrial Informatics (INDIN), 179-184.

[39] Kalyanmoy D., Agrawal S., Pratap A. and Meyarivan T. (2000). A fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. IEEE Transactions on Evolutionary Computation. 6(2), 182-197.

[40] Geem Z. W., Kim J. H., Loganathan G. V. (2001). A new heuristic optimization algorithm: harmony search. Simulation, 76(2), 60-68.

[41] Sivasubramani, S., Swarup, K. S. (2011). Multi-objective harmony search algorithm for optimal power flow problem. International Journal of Electrical Power & Energy Systems, 33(3), 745-752.

[42] Geem. Z. W. (2007). Harmony Search Algorithm for Solving Sudoku. Lecture Notes in Artificial Intelligence, 4692, 371-378.

[43] Geem, Z. W., Lee, K. S., Park, Y. (2005). Application of harmony search to vehicle routing. American Journal of Applied Sciences, 2(12), 1552-1557.

[44] Geem Z. W., Tseng C. L., Y. Park, C. L. (2005). Harmony Search for Generalized Orienteering Problem: Best Touring in China. Lecture Notes in Computer Science, 3412, 741-750.

[45] Landa-Torres, I. , Manjarres, D., Salcedo-Sanz, S., Del Ser, J., Gil-Lopez, S. (2013). A Multiobjective Grouping Harmony Search Algorithm for the Optimal Distribution of 24-hour Medical Emergency Units. Expert Systems with Applications, 40(6), 2343-2349.

[46] Bean, J. C. (1994). Genetic Algorithms and Random Keys for Sequencing and Optimization. ORSA Journal on Computing, 6(2), 154-160.

[47] Pirlot, M. (1996). General local search methods, European Journal of Operational Research, 2(3), 493-511.

[48] Rothlauf, F., Goldberg, D. E., Heinzl, A. (2002). Network random keysa tree representation scheme for genetic and evolutionary algorithms. Evolutionary Computation, 10(1), 75-97.

[49] Madeira, J. A., Pina, H. L., Rodrigues, H. C. (2010). GA topology optimization using random keys for tree encoding of structures. Structural and Multidisciplinary Optimization, 40(1-6), 227.

[50] Garcia-Santiago, C. A., Del Ser, J., Upton, C., Quilligan, F., Gil-Lopez, S., Salcedo-Sanz, S. (2015) A Random-Key encoded Harmony Search Approach for Energy-Efficient Production Scheduling with Shared Resources. Engineering Optimization, 47(11), 1481-1496.