Energy-Efficient Routing of a Multirobot Station: A Flexible Time-Space Network Approach

Jianbin Xin, *Member, IEEE*, Chuang Meng, Andrea D'Ariano, Frederik Schulte, Jinzhu Peng, *Member, IEEE*, and Rudy R. Negenborn

***Abstract*— This paper investigates a novel routing problem of a multi-robot station in a manufacturing cell. In the existing literature, the objective is to minimize the cycle time or energy consumption separately. The routing problem considered in this paper aims to reduce the cycle time and energy consumption jointly for each robot while avoiding collisions between these robots. For this routing problem, we propose a new flexible time-space network model that allows us to reduce energy consumption while minimizing the cycle time. The corresponding optimization problem is Mixed-Integer Nonlinear Programming (MINLP). For addressing its computational complexity, this paper designs a metaheuristic algorithm tailored to the studied problem and proposes an ε-constraint algorithm to study the trade-off between these two objectives. We conduct industrially relevant simulation experiments of case studies to show its effectiveness, in comparison to a conventional method, two state-of-the-art solvers, and two commonly-used metaheuristics. The results show that the proposed methodology can reduce energy consumption by up to 30% without compromising the cycle time. Meanwhile, the proposed algorithm can provide efficient solutions within a reasonable computation time.**

***Note to Practitioners*—This paper is motivated by the problem of improving energy efficiency when routing cooperative robots in a manufacturing station. In current approaches for routing multi-robot stations, the cycle time and energy consumption are minimized separately. This paper focuses on the movement of the robot end-effector and its connected joint and suggests a new approach to minimize these two objectives jointly by proposing a new mathematical model. The resulting planning problem is computationally intractable. A customized metaheuristic algorithm is thus designed for efficiently solving this planning problem. Our meta-heuristic algorithm is integrated with the ε-constraint method to study the relationship between these two objectives. Simulation experiments suggest that this approach can reduce energy consumption considerably, for the shortest cycle time, compared with the current approaches. In future research, the movements of multi-joints will be investigated whereby 3-D collision-free trajectory planning will be considered.**

***Index Terms*— Multi-robot systems, routing, flexible time-space network model, energy consumption, collision avoidance.**

Manuscript received 15 April 2022; revised 7 June 2022; accepted 17 July 2022. This article was recommended for publication by Editor Q. Zhao upon evaluation of the reviewers' comments. This work was supported in part by the National Natural Science Foundation of China under Grant 62173311 and Grant 61703372, in part by the College Youth Backbone Teacher Project of Henan Province under Grant 2021GGJS001, in part by the Henan Scientific and Technological Research Project under Grant 222102220123, in part by the Training Project of Zhengzhou University under Grant JC21640030, and in part by the Program for Science and Technology Innovation Talents in Universities of Henan Province under Grant 20HASTIT031. *(Corresponding author: Jinzhu Peng.)*

Jianbin Xin, Chuang Meng, and Jinzhu Peng are with the School of Electrical and Information Engineering, Zhengzhou University, Zhengzhou 450001, China (e-mail: j.xin@zzu.edu.cn; zzumengchuang@163.com; jzpeng@zzu.edu.cn).

Andrea D'Ariano is with the Department of Engineering, Roma Tre University, 00146 Rome, Italy (e-mail: a.dariano@dia.uniroma3.it).

Frederik Schulte and Rudy R. Negenborn are with the Department of Marine and Transport Technology, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: f.schulte@tudelft.nl; r.r.negenborn@tudelft.nl).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TASE.2022.3192914.

Digital Object Identifier 10.1109/TASE.2022.3192914

I. Introduction

ROBOTS considerably improve the productivity of manufacturing systems. In the production lines, a large number of industrial machines are operated collaboratively to perform various tasks [1], [2]. Typically, a robotic assembly line consists of several stations that are arranged serially or in parallel to reach production objectives [3]. At each station, multiple robots, often sharing the same workspace, work together to carry out complex operations (e.g., stud, welding, or sealing) within a predetermined period [4].

For a robotic manufacturing station, the primary goal is to minimize the cycle time. The cycle time (identical to the term *makespan* used in operations research) is a manufacturing indicator, indicating the total time from the start to the end of all the processes in the workstation. Producers aim to maximize the production rates to meet the increasing requirements made by customers. High production rates result from optimizing the intertwined operations of multiple robots within each station.

Besides the cycle time, producers regard energy consumption as another crucial objective to minimize [5]–[7]. As pointed out in [8], [9], a significant amount of energy is consumed by industrial robots in the manufacturing process, which is about 8 % of the total electrical energy consumed in production processes. Due to the existing strict policy guideline regarding CO_2 emissions and the rising energy price [9], reducing the robot energy consumption is a necessity. To maximize the profit, the producers expect to reduce robot energy consumption without deteriorating the service of high productivity, aiming for energy-efficient productions.

Motivated by the practical demand of improving energy efficiency when routing the cooperative robots in a manufacturing station, we investigate a new energy-efficient routing problem. In this problem, minimizing the cycle time and minimizing energy consumption are considered jointly. The corresponding optimization problem is not computationally tractable for practice-size scenarios, and an efficient algorithm will be developed to reduce its computational complexity.

A. Related Work

Related research on the planning of manufacturing cells has received increasing attention due to the development of 'Industry 4.0'. Here, two types of planning problems are generally identified: task allocation and routing problems.

As for a manufacturing system, the task allocation focuses on optimally distributing tasks to the available robots [10]. In [11], an intersection-free geometrical partitioning method is developed for the allocation of assembly tasks in a multi-robot cell. For improving the efficiency of multi-robot cells, the task allocation can also be introduced with the layout design simultaneously [12]. At the same time, the task allocation may need to be realized in a distributed way, because of the communication constraints and limitations [13].

In the routing problems, task orders and the routes of each robot need to be determined for executing different tasks in the robotic cell. For a high production rate, the main objective is to minimize the cycle time of all the robots, while satisfying the collision-free constraints for each robot. The integrated collision-free routing and scheduling problem of a multi-robot station was first studied in [4] using an iterative approach. Later, the task sequence and collision avoidance are considered simultaneously (in one framework) using a Time-Space Network (TSN) model, and the routing problem is regarded as a particular multiple Traveling Salesman Problem (mTSP) [14]. In these specific routing problems, the task is defined as a discrete action to perform a stud or welding operations by the manipulators.

Besides minimizing the cycle time, the minimization of energy consumption has recently received increasing attention when planning the routes in robotic cells. In [15], [16], the energy-aware scheduling of a single manufacturing robot has been studied to obtain the relation between the adjustable motion parameters and the energy consumption. Bukata *et al.* [8], [17] investigate the energy consumption problem by considering the robotic cell as a whole, proposing a parallel hybrid heuristic algorithm [8] and an exact parallel algorithm [17] to minimize the energy consumption following the desired cycle time which is not optimized. Furthermore, a cyclic multi-robot coordination problem of a press line is studied, and the energy consumption is minimized by smoothing the robot trajectories for the predefined paths [18].

The above research contributes to improving the efficiency of robotic manufacturing cells. The objectives of existing literature, however, aim at optimizing either the cycle time or the energy consumption individually. None of them considers minimizing these two objectives jointly.

B. Contributions

In this paper, we focus on minimizing the cycle time and energy consumption jointly for a robotic station, and this topic has not been carefully studied in the literature. The energy consumption here refers to the total amount of kinetic energy consumed by all robots to complete their tasks. The energy consumption is expected to be reduced, while maintaining a high production rate, thus meeting not only economic but also environmental criteria. To reach this simultaneous goal, we make the following contributions:

- We propose a new time-space network model for the energy-aware and collision-free routing problem of the multi-robot station. The proposed time-space network representation allows representing the motion time between successive tasks and task sequence as decision variables, resulting in a so-called Flexible Time-Space Network (FTSN) model to reduce energy consumption when performing the given tasks in the multi-robot station in a collision-free way. Existing TSN models of routing the multi-robot station regard the motion time as fixed parameters and energy consumption cannot be reduced [14]. Weighted-sum and lexicographic formulations are considered for determining the energy-efficient and collision-free route. Due to its nonlinear constraints and objective functions for minimizing energy consumption, a Mixed-Integer Nonlinear Programming (MINLP) problem remains to be solved.
- A customized metaheuristic algorithm is developed to efficiently solve the resulting MINLP problem based on the FTSN representation. For the studied MINLP, new two-dimensional encoding schemes and operators are proposed to deal with the mixed decision variables for tasks sequences and flexible motion times in a collision-free way. For the existing metaheuristic to solve the robot routing problem, the encoding scheme is typically limited to the one-dimension for sequence planning without avoiding collisions [19], [20]. Based on the customized metaheuristic, we further use the ε-constraint method to study the relationship between these two objectives.

As this paper builds on previous work such as [14], which addresses the cycle time minimization problem to decide the collision-free routes of a multi-robot station, it makes clearly different contributions. In [14], the motion times between every two successive tasks are regarded as fixed parameters. Thus, energy consumption concerning robot movements cannot be directly optimized. There, the optimization problem is formulated as Mixed-Integer Programming (MIP). In this work, an MINLP is proposed and solved due to nonlinear energy-related objectives and constraints. The resulting MINLP is more challenging to be solved than the MIP.

The remainder of this paper is organized as follows: Section II presents the mathematical formulation of the proposed FTSN model, for the considered energy-aware and collision-free routing problem. In Section III, a customized metaheuristic algorithm is designed to solve the corresponding MINLP problem. Section IV discusses and analyzes the simulation results carried out for industrial case studies. Section V concludes this paper and provides potential future research.

II. Flexible Time-Space Network Model

This section defines the considered problem of energy-efficient and collision-free routing of the multi-robot station and introduces the corresponding flexible time-space network formulation.

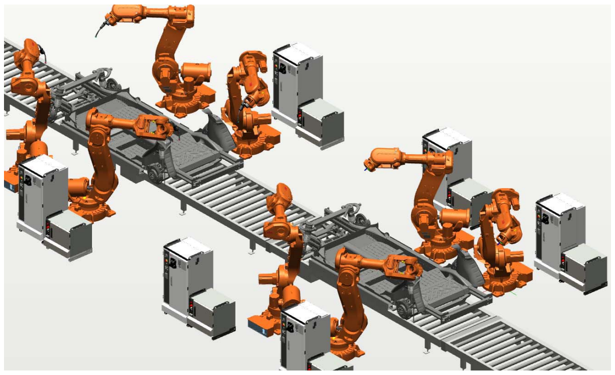


Fig. 1. Robotic assembly stations in the production line (demonstrated in software RobotStudio).

A. Problem Definition

At multi-robot stations of manufacturing systems, multiple robots are operated in a shared workspace to fulfill tasks that often require cooperation between the robots. Such settings can, for instance, be found in the automobile industry that uses cooperative robots in stud-welding or spot-welding, as illustrated in Fig. 1. For planning the collaborative motions of these robots, specific objectives and constraints need to be considered. A minimum cycle time is desired for maximizing productivity, but operating the robots at maximum speed may result in overly high energy consumption that stands in conflict with the economic and environmental policies of many operators. Moreover, since the robots share the workspace, collision avoidance must be granted.

Following the assumptions made in [14], the subsequent assumptions are made for the specific problem under consideration:

- All robots start from their idle positions.
- All robots are identical.
- A subset of tasks has been assigned to each robot in advance, but the task sequence for each robot needs to be decided. The task allocation can be obtained by solving a classification problem based on the position distribution of the nodes.
- Each task can be performed by one robot only. Each robot starts the next task after completing the current one.
- The workspace of the robots is considered in the same plane, and the tool center point is regarded as the reference point of each robot.
- The problem considers the operation of the end effector and its connected joint, disregarding the state of other joints. For this assumption, each robot could use the kinematic redundancy of the manipulator to obtain more degrees of freedom and resolve the conflict between different operation points which are nearby.

B. Flexible TSN Model

In the existing TSN model of the multi-robot routing problem [14], the motion time is considered as a fixed parameter, thus energy consumption cannot be further minimized. For achieving the energy-aware and collision-free routes of the multi-robot station, a flexible time-space network model that considers flexible motion times is proposed. This model has a nonlinear objective and adopts location constraints and time constraints.

TABLE I
NOMENCLATURE

Symbol	Description
V_k	Set of nodes assigned to robot k
E_k	Set of arcs that robot k may use
N_k	Number of tasks in V_k
m	Number of robots
k, k_1, k_2	Robot index, $k, k_1, k_2 \in \{1, 2, ..N\}$
i, j, n	Node index for a particular task, $i, j, n \in V_k$
T	Planning horizon
t	Time index, $t \in \{0, 1, .., T\}$
h_k	Idle position of robot k
$E_k^o(i)$	Set of arcs starting from node i for robot k
$E_k^s(i)$	Set of arcs ending at node i for robot k
$t_{ij}^{\min}$	Minimun time (integer) spent on arc (i, j)
d_{ij}	Distance of arc (i, j)
δ_{ik}	Given assignment of node i to robot k, $\delta_{ik} = 1$ if $i \in V_k$
(x_i, y_i)	Coordinate of node i
R	Diameter of the projected circular area occupied by the robot
M	A very large integer number

We assume that, for m robots, robot k ($k \in \{1, 2, .., m\}$) owns N_k tasks. In total, m directed graphs $G_k = (V_k, E_k)$ ($k \in \{1, 2, .., m\}$) are considered. For robot k, V_k ($V_k = \{u_1^k, u_2^k, \ldots, u_{N_k}^k\}$) is the collection of nodes and $E_k = \{(i, j) | i \in V_k, j \in V_k\}$ is the collection of arcs.

Each task is defined as executing a single operation (i.e., stud-welding and spot-welding) at a particular location. As a result, a specific node i corresponds to a place for the robot to perform a task. Arc (i, j) maps to the path from task i to task j. Nodes represent different locations for the robot to be visited by the robot.

We consider a planning horizon $T \times \Delta t$ equally discretized into a set of short time slots denoted by $\{\Delta t, 2\Delta t, \ldots, T \times \Delta t\}$. One time slot is given as Δt, and T is the total number of time slots. As a result, the TSN model can decompose the overall routing process of multiple robots into several time slots. At each time instant $t \in \{0, 1, .., T\}$, each robot can visit a particular node. Before detailing the proposed FTSN model, the used subscripts, parameters, and decision variables are introduced in Tables I and II.

For the flexible motion process of robot k on arc (i, j), the position update from $t-1$ to t is given as follows:

$$x_{kt} = x_{k(t-1)} + \sum_{i,j:(i,j \in E_k)} (u_{ijk(t-1)} \times \frac{x_j - x_i}{t_{ijk}}), \quad \forall k \tag{1}$$

$$y_{kt} = y_{k(t-1)} + \sum_{i,j:(i,j \in E_k)} (u_{ijk(t-1)} \times \frac{y_j - y_i}{t_{ijk}}), \quad \forall k \tag{2}$$

where t_{ijk} is a decision variable, and the motion time on arc (i, j) can be optimized. It is noted that equations (1)-(2) are both nonlinear equalities. In [14], t_{ijk} is regarded as a fixed parameter and the motion time of robot k on arc (i, j) cannot be further changed. Fig. 2 illustrates the discretized motion process on arc (i, j), and this process depends on t_{ijk}. As t_{ijk} is modified, the number of fictitious points also changes.

The time connectivity of robot k for arc (i, j) is modelled based on the cumulative flow variables a_{ijkt} and d_{ijkt}. This

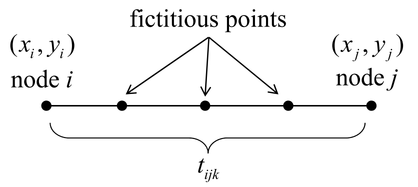


Fig. 2. Discretized flexible motion process on arc (i, j) depending on the decision variable t_{ijk}.

TABLE II
DECISION VARIABLES

Variable	Description
z_{ijk}	0-1 variable, $z_{ijk} = 1$ means that robot k moves over the arc (i, j), otherwise it is 0
a_{ijkt}	0-1 variable, $a_{ijkt} = 1$ when robot k reaches the arc (i, j) at time t, otherwise it is 0
d_{ijkt}	0-1 variable, robot k is 1 when leaving arc (i, j) at time t, otherwise 0
u_{ijkt}	0-1 variable, robot k is 1 when occupying arc (i, j) at time t, otherwise 0
$c^q_{k_1k_2t}$	0-1 variable, collision avoidance coefficient between robots k_1 and k_2 at time t
t_{ijk}	Motion time spent on arc (i, j) for robot k (integer variables)
x_{kt}	Real variable, x-coordinate of robot k at time t
y_{kt}	Real variable, y-coordinate of robot k at time t

connectivity is described by the following constraints:

$$t_{ijk} \times z_{ijk} = \sum_t (t \times (d_{ijkt} - d_{ijk(t-1)})) - \sum_t (t \times (a_{ijkt} - a_{ijk(t-1)})), \quad \forall k, (i, j) \in E_k \tag{3}$$

$$a_{ijkt} \geq a_{ijk(t-1)}, \quad \forall k, (i, j) \in E_k, t \tag{4}$$

$$d_{ijkt} \geq d_{ijk(t-1)}, \quad \forall k, (i, j) \in E_k, t \tag{5}$$

$$u_{ijkt} = a_{ijkt} - d_{ijkt}, \quad \forall k, (i, j) \in E_k, t \tag{6}$$

$$\sum_{i.j:(i,j\in E_k)} u_{ijkt} \leq 1, \quad \forall k, t \tag{7}$$

$$t_{ijk} \geq t^{\min}_{ij}, \quad \forall k, (i, j) \in E_k \tag{8}$$

where equality (3) is the time constraint for each robot between two successive tasks, and each robot can wait for more than one time unit in the node. Note that Inequality (3) is a nonlinear constraint due to the term $z_{ijk} \times t_{ijk}$. Inequalities (4)-(5) are the time connectivity constraints for robot k to perform two successive tasks regarding the arrival and departure for arc (i, j). Inequalities (6)-(7) guarantee that each robot can only perform one task per time unit. inequality (8) ensures that the time spent on arc (i, j) for robot k is no less than the minimum time ($t^{\min}_{ij} = 1$ if $i = j$).

Additional constraints are needed to map the time and space in the same framework as follows:

$$\sum_{i:(i,h_k)\in E_k} d_{ih_kkT} \geq 1, \quad \forall k \tag{9}$$

$$\sum_{j:(h_k,j)\in E_k} a_{h_kjkt} = 1, \quad \forall k \tag{10}$$

$$z_{ijk} = a_{ijkT}, \quad \forall k, (i, j) \in E_k \tag{11}$$

$$\sum_{i,j:(i,j)\in E_k} d_{ijkt} = \sum_{j.n:(j,n)\in E_k} a_{jnkt} \quad \forall k, j \in V_k - h_k \tag{12}$$

$$d_{ijkt} \leq a_{ijkt}, \quad \forall k, (i, j) \in E_k, t \tag{13}$$

where inequalities (9)-(10) are time constraints of each robot at the start and the end positions, ensuring that each robot moves from the start of the planning horizon. Constraint (11) is the mapping constraint between the time-space network and the physical routing network. Constraints (12)-(13) guarantee the continuity of the arrival and departure times for robot k when visiting arc (i, j).

Furthermore, the constraints for assigning the tasks of each robot are given as follows:

$$\sum_{i,j:(i,j)\in E^{o}_k(h_k)} z_{ijk} \geq {}^1, \quad \forall k \tag{14}$$

$$\sum_{i:(i,j)\in E^{s}_k(j)} z_{ijk} = \sum_{n:(j.n)\in E^{o}_k(j)} z_{jnk}, \quad \forall k, \ j \in V_k - h_k \tag{15}$$

$$\sum_{i,j:(i,j)\in E^{s}_k(h_k)} z_{ijk} \geq 1, \quad \forall k \tag{16}$$

$$\sum_{k=1} \sum_{j=1} z_{ijk} \geq 1, \quad \forall i \tag{17}$$

$$z_{ijk} \leq \delta_{ik}, \quad \forall i, k, (i, j) \in E_k \tag{18}$$

$$z_{jik} \leq \delta_{ik}, \quad \forall i, k, (j, i) \in E_k \tag{19}$$

where constraints (14)-(16) ensure the task sequence at the start position, intermediate position, and end position of each robot. Constraint (17) guarantees that all tasks are executed. Constraints (18)-(19) ensure that arc (i, j) can only be visited by robot k when task i is assigned to it.

The collision-free constraints for any two robots are provided to guarantee that at any time t each robot does not collide with the other robot, as suggested in [21]. These constraints are given as follows:

$$x_{k_1t} - x_{k_2t} \geq R - Mc^1_{k_1k_2t}, \quad \forall k_1, k_2, \quad \forall t \tag{20}$$

$$x_{k_2t} - x_{k_1t} \geq R - Mc^2_{k_1k_2t}, \quad \forall k_1, k_2, \quad \forall t \tag{21}$$

$$y_{k_1t} - y_{k_2t} \geq R - Mc^3_{k_1k_2t}, \quad \forall k_1, k_2, \quad \forall t \tag{22}$$

$$y_{k_2t} - y_{k_1t} \geq R - Mc^4_{k_1k_2t}, \quad \forall k_1, k_2, \quad \forall t \tag{23}$$

$$\sum_{q=1}^{4} c^q_{k_1k_2t} \geq 3, \quad \forall k_1, k_2, \quad \forall t \tag{24}$$

where (x_{k_1t}, y_{k_1t}) and (x_{k_2t}, y_{k_2t}) are the coordinates of the end-effectors for robots k_1 and k_2 at time t. These coordinates are computed using formulas (1)-(2). $c^q_{k_1k_2t}$ is the decision variable for avoiding the collisions. $c^q_{k_1k_2t} = 0$ indicates that there is at least R distance between the end-effectors of robots k_1 and k_2 in the q^{th} direction in the plane, while $c^q_{k_1k_2t} = 1$ means that the related constraint is relaxed. These constraints ensure that the end-effector ranges of any two robots avoid collisions either at a node or at a fictitious point of the arc for each robot, as illustrated in Fig. 3. However, an industrial robot is a 3D entity, and the constraints (20)-(24) cannot fully guarantees the collision avoidance of the whole body and every joint, as we focus on the end-effector and its connected joint.

C. Objective Function

Here, we give the robot routing problem in the weighted-sum and the lexicographical formulations. These

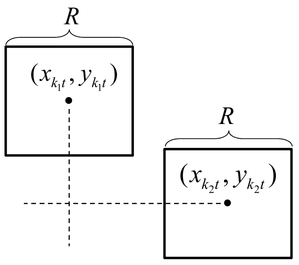


Fig. 3. Illustration of collision avoidance for the end-effector areas.

formulations optimize the multiple objectives with different priorities. For our robot routing problem, we consider two objectives: the cycle time (denoted by F_c) and the kinetic energy consumption (represented by F_e). To meet the requirement from the manufacturer, minimizing the cycle time is prioritized, while minimizing the energy consumption is less critical. F_c and F_e are formulated as follows:

$$F_c = \max\{\sum_t t \times \sum_{i:(i,h_k)\in E_k^s(h_k)} [d_{ijkt} - d_{ijk(t-1)}], \quad \forall k\} \quad (25)$$

$$F_e = \sum_k \sum_{i.j:(i,j)\in E_k} p_{ij}^1 \times t_{ijk} + p_{ij}^2 + p_{ij}^3 \times t_{ijk}^{-1} \quad (26)$$
$$+ p_{ij}^4 \times t_{ijk}^{-2} + p_{ij}^5 \times t_{ijk}^{-3}.$$

Formula (25) gives the cycle time of all robots as defined in [14]. Formula (26) represents the energy consumption formulation of all the robotic motions required to complete the assigned tasks ($(i, j) \in E_k$) at the planning level. Formula (26) models the kinetic energy consumption of the robotic motion from node i to node j by a non-linear function of the travel time t_{ijk} for robot k [22]. This formulation enables the integration of the robot energy optimization into scheduling these motions. In (26), $p_{ij}^1, \ldots, p_{ij}^5$ represent five constant parameters related to kinetic energy from node i to node j. The details of formula (26) are well explained in [22].

First, we use the weighted-sum formulation, in which F_c and F_e are represented via a linear combination and optimized simultaneously. The corresponding overall optimization problem, defined as P_1, is described as follows:

$$P_1: \quad \min(F_c + \lambda F_e)$$
$$\text{s.t.} \quad (1) - (24),$$

where λ is the weighting factor and its unit is normalized.

Note that F_e is a nonlinear function according to (26), and constraints (1), (2), and (3) are also nonlinear. When simultaneously optimizing the cycle time and the kinetic energy consumption, the decision variables listed in Table II are binary or integer. Therefore, an MINLP model is needed. Now we prove that P_1 is a non-convex MINLP.

Remark 1: The optimization problem P_1 is a non-convex nonlinear optimization problem.

Proof: It is easy to prove that $t_{ijk}z_{ijk}$ in constraint (3) is a nonlinear function, since t_{ijk} and z_{ijk} are both decision variables.

The Hessian matrix of $t_{ijk}z_{ijk}$ is computed as $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, and its determinant is negative. Therefore, the function $t_{ijk}z_{ijk}$ is a non-convex function.

Considering $t_{ijk}z_{ijk}$ is independent to a_{ijkt} and d_{ijkt}, constraint (3) is a nonlinear and non-convex constraint.

The MINLP problem P_1 has at least one non-convex constraint. Thus we conclude that P_1 is a non-convex MINLP. □

In addition to the weighted-sum strategy, we also consider the lexicographical strategy, in which the cycle time F_c is regarded as the primary objective while the energy consumption F_e is regarded as the secondary objective. The lexicographical formulation of our optimization problem, denoted as P_2, is given as follows:

$$P_2: \quad \min F_e$$
$$\text{s.t.} \; F_c \leq F_c^*, \; \text{and (1)-(24)},$$

where F_c^* denotes the minimal cycle time subject to constraints (1)-(24). Similar to Remark 1, the problem P_2 can be proved to be a non-convex MINLP.

In general, non-convex MINLPs are more difficult to solve than the MIP which owns linear constraints and objectives. In the next section, efficient algorithms are developed to solve the formulated MINLPs (P_1 and P_2), thus obtaining collision-free and energy-efficient routes for multiple robots.

III. Solution Algorithm

In this section, we design a customized Genetic Algorithm (GA) to efficiently solve the MINLP problems formulated in Section II. MINLP problems are known to be NP-hard [23], and commercial MINLP solvers, like BARON, cannot provide high-quality solutions in a reasonable computation time. Therefore, an efficient algorithm is demanded to solve the formulated MINLP problems based on the FTSN formulation.

GAs have been proved to be useful for successfully solving complex combinatorial problems (such as MIP and MINLP) [24]–[28]. GA has a relatively simple algorithmic structure, but this metaheuristic has a good ability to diversify the search in the feasible region of the search space [29]. Our studied routing problem is a variant of the multiple Traveling Salesman Problem (mTSP), and the GA is efficient for solving such a problem [30]. Here, for addressing the MINLP in this paper, we design new encoding schemes and operators, allowing the customized GA to be suitable for dealing with the considered MINLP problems. Based on the customized GA, we design an ε-constraint method to study the relationship between the two objectives.

A. Encoding

In our GA, the encoding for the population (the set of candidate solutions) needs to be initialized. The encoding scheme for each solution is highly relevant to the solution quality. For the routing problem of the multi-robot station, the encoding scheme of existing GAs is designed for the solution to MIPs only [14], and this scheme cannot be used for MINLPs. In this part, we develop a new encoding scheme for constructing the solutions suitable for the considered MINLP.

$$\begin{array}{ll}
\text{Dimension1} & h_k \ldots h_k, u_i^k \ldots u_i^k, u_j^k \ldots u_j^k, \quad \ldots \quad , u_n^k \ldots u_n^k \\
\text{Dimension2} & \underbrace{u_i^k \ldots u_i^k}_{t_{h_k ik}}, \underbrace{u_j^k \ldots u_j^k}_{t_{ijk}}, \quad \ldots \quad , u_n^k \ldots u_n^k, h_k \ldots h_k
\end{array}$$

Fig. 4. Illustration of composing the two dimensional $X_{\mathrm{s},k}$ for robot k.

Given m robots and a task set V_k for robot k ($k \in \{1, 2, ..m\}$). $V = V_1 \cup V_2 \ldots \cup V_m$ is the set of all tasks. For robot k, the set $V_k = \{u_1^k, u_2^k, \ldots, u_{N_k}^k\}$ and $u_i^k \neq u_j^k$ $(i \neq j)$

We consider a mixed encoding scheme. The solution X contains X_{t} and X_{s}, representing two types of information (motion time and task sequence, respectively). X_{t} represents a matrix containing t_{ijk} (the motion time spent on each arc) of all robots ($k = 1, 2, \ldots m$) to complete all required tasks. The detailed composition of X_{t} is represented as follows:

$$X_{\mathrm{t}} = \begin{bmatrix} M_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & M_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & M_m \end{bmatrix}, \tag{27}$$

where the formulation of M_k ($k = 1, 2, \ldots m$) is given as follows:

$$M_k = \begin{bmatrix} t_{u_1^k u_1^k k} & \cdots & t_{u_{N_k}^k u_1^k k} \\ \vdots & \ddots & \vdots \\ t_{u_1^k u_{N_k}^k k} & \cdots & t_{u_{N_k}^k u_{N_k}^k k} \end{bmatrix}. \tag{28}$$

X_{s} follows a two-dimensional encoding scheme for sequencing the collision-free routes of robots, as shown in Fig. 4. X_{s} is made of the following parts: $X_{\mathrm{s},1}$, $X_{\mathrm{s},2}, \ldots, X_{\mathrm{s},m}$ for each robot. Dimensions 1 and 2 denote the start and end nodes within several time slots that correspond to the time spent between these nodes. As seen from Fig. 4, u_i^k in Dimension 1 and u_j^k in Dimension 2 are the start and end nodes of the visited arc (u_i^k, u_j^k) for which t_{ijk} is spent by robot k.

The encoding length for each robot is the planning horizon T, and the total length of the encoding scheme is mT. Each column of the encoding is a time slot indicating that the robot is occupying a particular path within this time window. More details on X_{s} are explained in Section III of [14]. We noted that the number of columns for robot k, occupied by arc (i, j), corresponds to the (integer) number t_{ijk}, which is described in the elements of M_k. The initialization of the solution X is provided in the paper appendix.

B. Energy-Efficient Algorithm

1) Choices of Operators: In this part, we detail the customized operators of the developed GA for solving the MINLPs based on the collision-free mTSP presented in Section II. The strategies used for the selection and mutation are constructed to customize the genetic algorithm to efficiently solve the mTSP. The routing problem investigated in our manuscript is a variant of the mTSP, and the TSP is its fundamental version. When using the GA to solve the TSP problem, the removal of the crossover operator is suggested by the observation that the crossover could result in longer computation times and deteriorate the solution quality [31]. The mutation operator is regarded as the most effective one when solving the TSP, since mutation prevents the algorithm to be trapped in a local minimum [31]. Therefore, only the selection and mutation operators are considered in our developed GA.

Regarding the selection strategy, the top-ranking method is employed. At each iteration, the best 1/8 solutions are selected from the entire population (as the elite solutions) and retained until the next iteration.

Our mutation strategy considers seven operators to deal with the mixed decision variables (motion time and task sequence). These seven operators are based on three general operators, namely, *flip, swap*, and *slide*. These are considered in the proposed algorithm. The flip mutation works by randomly choosing two positions in the chromosome and reversing the order in which their values appear between those positions. The swap operator randomly swaps the values of two positions in the chromosome. For the slide operator, two positions in the chromosome are randomly selected, and the contents of these two positions move one position to the left. These three operators can be described as follows [31]:

$$flip(\pi, p_1, p_2) \triangleq \pi'(p_1 : p_2) = \pi(p_1, -1, p_2), \tag{29}$$

$$swap(\pi, p_1, p_2) \triangleq \pi'(p_1) = \pi(p_2), \pi'(p_2) = \pi(p_1), \tag{30}$$

$$slide(\pi, p_1, p_2) \triangleq \pi'(p_1 : p_2) = [\pi(p_1 + 1 : p_2), \pi(p_1)], \tag{31}$$

where π is a segment of the solution. π is a permutation, and π can be X_{t} and X_{s}. p_1 and p_2 represent two positions of the segment. Since we use a two-dimensional encoding scheme for representing the task sequence X_{s}, the positions of two columns are selected for performing the flip, swap, and slide operations. As for the motion time part, the contents of two rows of X_{s} are selected for the proposed mutation operations.

2) Main Procedures: Following the developed encoding scheme, we now present the main procedures of the customized GA. Algorithm 1 gives the pseudocode to solve the considered MINLP problem. As discussed above, flip, swap, and slide operators are the general mutation operators used by the customized GA. Since two types of decision variables (task sequence and motion time) are involved, the flip, swap, and slide operations are performed for the single X_{s} and combined X_{s} and X_{t}. In addition to the above operators, we also consider a random operator to generate motion times for X_{t}. The details of these operations are given in Lines 13-19 of Algorithm 1.

Table III lists the related notations used in the developed algorithm. Sub-populations $P_1(i_{\mathrm{iter}})$–$P_8(i_{\mathrm{iter}})$ constitute the entire population $P(i_{\mathrm{iter}})$. The sizes of $P_1(i_{\mathrm{iter}})$–$P_8(i_{\mathrm{iter}})$ are all assumed to be one eighth of the entire population $P(i_{\mathrm{iter}})$ $(N_{\mathrm{p}}/8)$. $P_1(i_{\mathrm{iter}})$ is the elite sub-population, while $P_2(i_{\mathrm{iter}})$–$P_8(i_{\mathrm{iter}})$ are the sub-populations of seven operations based on $P_1(i_{\mathrm{iter}})$. We use the notations $X_{\mathrm{s}}(i_{\mathrm{iter}}, i_{\mathrm{p}})$ and $X_{\mathrm{t}}(i_{\mathrm{iter}}, i_{\mathrm{p}})$ to represent the i_{p}th elite solutions of $P_{X_{\mathrm{s}}}^1$ and $P_{X_{\mathrm{t}}}^1$ that constitute $P_1(i_{\mathrm{iter}})$ at iteration i_{iter}.

Algorithm 1 Weighted-Sum GA for the Formulated MINLP

```
1: $i_{\text{iter}} = 0$
2: initialize $P(i_{\text{iter}})$
3: **if** $P(i_{\text{iter}})$ has infeasible solutions **then**
4:     repair $P(i_{\text{iter}})$
5: **end if**
6: **while** $i_{\text{iter}} \leq i_{\max}$ **do**
7:     **for** **do**$i_{\text{p}} = 1$ to $N_{\text{p}}$
8:         evaluate the fitness of $F(X)$ as $P(i_{\text{iter}})$
9:     **end for**
10:    select 1/8 of $P(i_{\text{iter}})$ with the lower fitness as $P_1(i_{\text{iter}})$
11:    **for** $i_{\text{p}} = 1$ to $N_{\text{p}}/8$
12:        $k = randi(m)$
13:        flip operation to the $k$th part of $X_{\text{s}}(i_{\text{iter}}, i_{\text{p}})$ used for $P_2(i_{\text{iter}}, i_{\text{p}})$, no change to $X_{\text{t}}(i_{\text{iter}}, i_{\text{p}})$
14:        swap operation to the $k$th part of $X_{\text{s}}(i_{\text{iter}}, i_{\text{p}})$ used for $P_3(i_{\text{iter}}, i_{\text{p}})$, no change to $X_{\text{t}}(i_{\text{iter}}, i_{\text{p}})$
15:        slide operation to the $k$th part of $X_{\text{s}}(i_{\text{iter}}, i_{\text{p}})$ used for $P_4(i_{\text{iter}}, i_{\text{p}})$, no change to $X_{\text{t}}(i_{\text{iter}}, i_{\text{p}})$
16:        rand operation to $X_{\text{t}}(i_{\text{iter}}, i_{\text{p}})$ as part of $P_5(i_{\text{iter}}, i_{\text{p}})$, no change to $X_{\text{s}}(i_{\text{iter}}, i_{\text{t}})$
17:        flip operation to the $k$th part of $X_{\text{s}}(i_{\text{iter}}, i_{\text{p}})$ and the whole part of $X_{\text{t}}(i_{\text{iter}}, i_{\text{p}})$ used for $P_6(i_{\text{iter}}, i_{\text{p}})$
18:        swap operation to the $k$th part of $X_{\text{s}}(i_{\text{iter}}, i_{\text{p}})$ and the whole part of $X_{\text{t}}(i_{\text{iter}}, i_{\text{p}})$ used for $P_7(i_{\text{iter}}, i_{\text{p}})$
19:        slide operation to the $k$th part of $X_{\text{s}}(i_{\text{iter}}, i_{\text{p}})$ and the whole part of $X_{\text{t}}(i_{\text{iter}}, i_{\text{p}})$ used for $P_8(i_{\text{iter}}, i_{\text{p}})$
20:    **end for do**
21:    $P(i_{\text{iter}}+1) = P_1(i_{\text{iter}}) \cup P_2(i_{\text{iter}}) \cup P_3(i_{\text{iter}}) \cup P_4(i_{\text{iter}}) \cup P_5(i_{\text{iter}}) \cup P_6(i_{\text{iter}}) \cup P_7(i_{\text{iter}}) \cup P_8(i_{\text{iter}})$
22:    $i_{\text{iter}} = i_{\text{iter}} + 1$
23:    **if** $P(i_{\text{iter}})$ has infeasible solutions **then**
24:        repair $P(i_{\text{iter}})$
25:    **end if**
26: **end while**
```

For our customized GA, all the constraints (1)-(24) must be satisfied. For this, proper measures should be taken when designing this metaheuristic. Regarding the generated infeasible solutions that do not satisfy constraints (1)-(19), we use repairing operations. For any two successive arcs, the end node of the first arc must be the start node of the next arc. Also, for each arc, the number of columns must be consistent with its motion time. If t_{ijk} is below its minimum value for Constraint (8), t_{ijk} will be replaced by its minimum value.

Regarding the remaining constraints, a penalty function, defined as $p(i_{\text{iter}}, X)$, is included in the fitness function $F(X)$, together with the objective function defined as $F_0(X)$. The composition of $F(X)$ is thus given as follows:

$$F(X) = F_0(X) + p(i_{\text{iter}}, X) \tag{32}$$

where $F_0(X)$ represents the weighted sum of cycle time and energy consumption for solution X, defined by following the objective function of P_1 as follows:

$$F_0(X) = F_{\text{c}}(X) + \lambda F_{\text{e}}(X). \tag{33}$$

TABLE III

NOTATIONS USED FOR THE DESIGNED ALGORITHM

Notation	Description
i_{iter}	Iteration index
$i_{\max}$	Maximum number of iterations
N_{p}	Population size
i_{p}	Solution index of the population
X	Set of solutions
$F(X)$	Fitness function of solution set X
$P(i_{\text{iter}})$	Entire population at iteration i_{iter}
$P_1(i_{\text{iter}})$	Sub-population used for elite at iteration i_{iter}
$P_2(i_{\text{iter}}) \sim P_8(i_{\text{iter}})$	Sub-populations of seven operations at iteration i_{iter}
$P_2(i_{\text{iter}}, i_{\text{p}}) \sim P_8(i_{\text{iter}}, i_{\text{p}})$	i_{p}th solution of $P_2(i_{\text{iter}}) \sim P_8(i_{\text{iter}})$
$X_{\text{t}}(i_{\text{iter}}, i_{\text{p}})$	i_{p}th elite solution of $P^1_{X_{\text{t}}}$ at iteration i_{iter}
$X_{\text{s}}(i_{\text{iter}}, i_{\text{p}})$	i_{p}th elite solution of $P^1_{X_{\text{s}}}$ at iteration i_{iter}

To satisfy the collision-free constraints (20)−(24), a penalty function $p(i_{\text{iter}}, x)$ for iteration i_{iter} is defined as follows:

$$p(iter, X) = (\rho_{i_{\text{iter}}})^{\alpha}(d_1(X)^{\beta} + d_2(X)^{\beta}), \tag{34}$$

where $\rho_{i_{\text{iter}}}$ is a variable multiplication factor, $d_1(X)$ is a function to penalize the solution that fails to satisfy constraints (20)−(24), and $d_2(X)$ is a function to punish the solution that does not respect the motion time constraint (8). α and β are the parameters that adjust the size of the penalty value.

The values of $\rho_{i_{\text{iter}}}$, $d_1(X)$, and $d_1(X)$ are designed as follows:

$$\rho_{i_{\text{iter}}} = C i_{\text{iter}} \tag{35}$$

$$d_1(X) = \begin{cases} 0, & X \text{ is feasible} \\ |l - R|, & \text{otherwise} \end{cases} \tag{36}$$

$$d_2(X) = \begin{cases} 0, & X \text{ is feasible} \\ \left|(t_{ijk} - t^{\min}_{ijk})\right|, & \text{otherwise} \end{cases} \tag{37}$$

where C is a constant, l is the minimum distance between two arbitrary robots on all time unit windows, and R is the minimum safety distance. t_{ijk} is the actual motion time for robot k on arc (i, j).

In addition to the weighted-sum strategy, we also consider a lexicographic strategy. In the lexicographic strategy, the cycle time is regarded as the primary objective, while the energy consumption is treated as the secondary one. The minimal cycle time is obtained by solving a single-objective optimization problem; the obtained cycle time is then incorporated into Algorithm 1 as an additional constraint. These two strategies will be evaluated in Section IV.

C. Algorithm for the Pareto Frontier Analysis

This section proposes an algorithm for computing the Pareto frontier for the cycle time and energy consumption minimization by using the ε-constraint method. The ε-constraint method can be efficiently used for computing non-dominated solutions [32], [33]. Here, we propose an iterative procedure for a framework based on the ε-constraint method and the developed GA. At each iteration, we solve a single-objective formulation for the studied optimization problem. In this problem, one performance indicator is optimized directly in the objective function, while the other performance indicator is indirectly

Algorithm 2 ε-Constraint Method

```
1: $i_{\text{iter}} = 0$
2: min $F_c$ subject to constraints (1)-(24), and set $\beta_1 = F_c^*(i_{\text{iter}})$, $\varphi_2 = F_e''(i_{\text{iter}})$
3: min $F_e$ subject to constraints (1)-(24), and set $\beta_2 = F_e^*(i_{\text{iter}})$
4: insert the pair ($\beta_1$,$\varphi_2$) in the Pareto solution set $\Psi$
5: while $\beta_2 < \varphi_2$ do
6:     $i_{\text{iter}} = i_{\text{iter}} + 1$
7:     min $F_c$ subject to constraints (1)-(24) plus the constraint: $F_e < \varphi_2$, and set $\beta_1 = F_c^*(i_{\text{iter}})$, $\varphi_2 = F_e''(i_{\text{iter}})$
8:     insert the pair ($\beta_1$,$\varphi_2$) in the Pareto solution set $\Psi$
9: end while
10: return the Pareto solution set $\Psi$
```

optimized by inserting an additional bound constraint in the single-objective formulation.

Algorithm 2 describes the main steps of our ε-constraint method. At each iteration i_{iter}, the values of $F_c^*(i_{\text{iter}})$ and $F_e^*(i_{\text{iter}})$ are computed individually. $F_e''(i_{\text{iter}})$ denotes the value of F_e regarding the optimal solution of F_c^* at iteration i_{iter}. The proposed ε-constraint method initializes the number of iterations i_{iter} and finds the optimal values of the two single-objective optimization problems. The value of the optimal single-optimization solution (β_1, φ_2) is inserted into the solution set Ψ. After this initialization, a new single-optimization problem is iteratively solved by adding an additional constraint on the value of the secondary indicator F_e. For each iteration, a new solution pair (β_1, φ_2) is added into the solution set Ψ. When the value of $F_e''(i_{\text{iter}})$ is equal to $F_e^*(i_{\text{iter}})$, the iterative process ends and returns the set Ψ.

IV. Case Studies

This section discusses the computational results obtained from the simulation experiments carried out to demonstrate the effectiveness of the proposed methodology for planning the collision-free and energy-aware routing of multiple robots at one workstation. The simulation settings are provided, and several case studies in the automotive industry are then conducted.

A. Simulation Settings

To study the effectiveness of the proposed methodology, two typical types of case studies (spot welding on a car door and spot welding on a car underbody) are considered [14]. The settings of our case studies are given in Table IV, in which seven scenarios are included. Scenarios 1-2 and Scenarios 3-7 are considered for the first type and for the second type of case studies, respectively. For each scenario, the operation nodes are distributed equally, and ten experiments have been carried out. These settings are suggested by [4], [14]. The maximum computation time is set to 1 hour.

Based on the proposed FTSN model, the developed weighted-sum GA and lexicographic GA (WGA and LGA for short) are compared with a conventional method and two state-of-the-art MINLP solvers (BARON and SCIP). The latter three methods are explained shortly as follows:

TABLE IV
Setting of the Considered Case Studies

Description	Robots	Tasks	Object
Scenario 1	2	4	Door
Scenario 2	2	10	Door
Scenario 3	2	10	Underbody
Scenario 4	2	20	Underbody
Scenario 5	4	20	Underbody
Scenario 6	4	30	Underbody
Scenario 7	4	40	Underbody

- The conventional method is based on the TSN model considering fixed motion times and on the GA developed in [14] to solve the corresponding MIP that minimizes the cycle time only.
- BARON is a commercial solver used to solve MINLP problems [34], [35]. The cycle time and the energy consumption are minimized in a weighted-sum form when solving the considered MINLP.
- SCIP is the fastest non-commercial solver to solve the MINLP [36]. The problem formulation for the solver SCIP is the same MINLP as for the solver BARON.

Both BARON and SCIP implement a spatial branch and bound algorithm that utilizes linear programming for the bounding step to solve MINLP problems. The algorithm in the solver BARON is enhanced by using advanced box reduction techniques and convexification techniques for quadratic functions. Further, BARON uses NLP relaxations for bounding. These improvements are not considered in the solver SCIP [37].

In addition to the three methods above, two commonly-used metaheuristics (Tabu Search (TS) and Variable Neighborhood Search (VNS)) are evaluated for further comparison. The tested TS and VNS are next briefly described:

- Tabu search is a deterministic metaheuristic based on local search. The implemented TS follows the standard algorithmic procedures presented in [38]. The tabu list is used to escape from the local optimum. Initially, this list is empty. The encoding is set the same as the proposed GA. For generating the candidate list (similarly to the GA population), the neighborhood solutions of a candidate solution are modified by adopting one of the seven operators presented in Algorithm 1 (Lines 13-19) randomly chosen, and the probability for selecting each operation is equal. Regarding the aspiration criterion, if the tabu list contains all the seven operations, the operation with the best solution is removed from the tabu list. Here, no advanced intensification or diversification strategies are used.
- The implemented VNS is the standard version of VNS [39], which combines deterministic and random changes in neighborhoods. This algorithm consists of two phases: the shaking phase for the global search and the improvement phase for the local search. The encoding is the same as the proposed GA to construct the neighborhood. For the global search, one of the seven operations presented in Algorithm 1 (Lines 13-19) is randomly selected. For the local search, the seven operations in Algorithm 1 (same as above) are selected to modify the neighborhood solutions by following the

TABLE V
AVERAGED PERFORMANCE FOR SCENARIOS 1–3 VIA WGA

N_p	i_{max}	F_c (Seconds)	F_e (KJ)	Computation time (Seconds)
90	100	10	6.17	9.0
90	200	10	6.13	19.2
90	300	10	6.11	26.7
120	100	10	6.16	10.7
120	200	10	6.09	21.4
120	300	10	6.08	31.0
150	100	10	6.11	11.2
150	200	10	6.09	23.5
150	300	10	6.08	34.9

TABLE VI
COMPUTATIONAL RESULTS WHEN VARYING λ

λ	Scenario 1		Scenario 2	
	F_c (Seconds)	F_e (KJ)	F_c (Seconds)	F_e (KJ)
10^{-2}	9	1.10	13	2.15
10^{-3}	5	3.22	8	6.24
10^{-4}	5	3.22	7	7.12
10^{-5}	5	3.22	7	7.12

sequencing strategy in Algorithm 1. The local search stops when the first improvement is achieved.

For the proposed GA, the population size N_p and the maximum number of iterations i_{max} are important parameters. We thus conducted a group of numerical experiments for Scenarios 1-3 via the developed WGA to better set these parameters, as presented in Table V. In these experiments, the parameter λ is set to a very small positive number to prioritize the makespan. Table V gives the average cycle time, the energy consumption, and the computation time for these experiments. The chosen N_p and i_{max} are 120 and 200, because this setting reaches a good balance between the solution quality and the computational effort. The settings of $N_p = 120$ and $i_{max} = 300$ are also a good choice, but the computation time increases by about 50% with a very small reduction in F_e when compared to the settings of $N_p = 120$ and $i_{max} = 200$.

With the above settings of N_p and i_{max}, we study the effect of varying the weighting factor λ on the cycle time and energy consumption. Table VI compares the values of these performance indicators for Scenarios 1 and 2. As λ becomes large, the energy consumption reduction becomes relatively more critical, therefore more energy can be saved while the cycle time grows. To highlight the importance of the cycle time, a very small positive value of λ is needed. To guarantee a considerable prioritization of the cycle time over the energy consumption, λ is set as 10^{-5}.

Regarding the parameters of the penalty function used to avoid collisions, C is set to 0.5, and α and β are both set to be 2. The values of these parameters are the same as the ones suggested in [14] since the same safety distance is considered.

The maximum computation time for all the metaheuristics (GA, TS, and VNS) is set to 600 seconds. The motion time t_{ijk} for the rand operator is randomly generated between $[t_{ijk}^{\min}, 2t_{ijk}^{\min}]$. For the lexicographic strategy of GA, VNS, and TS, the MaxFEs to individually compute the two objectives is set to half of the weighted-sum value.

The hardware for all simulations is an Intel i7-9700 processor (3.0GHz) with 8GB of memory. The optimization problems are modelled and solved in Matlab R2018. The software ABB RobotStudio is used to simulate the robot operations and verify the results obtained via the FTSN model. RobotStudio is a state-of-the-art robot simulation software, which copies the real software that moves the robots into a production environment. This simulation software allows to perform realistic simulations and a careful assessment of the optimized solutions, using real robot programs and configuration files identical to those used on the shop floor [40]. In this software, the IRB2400 series robot is selected.

TABLE VII
CYCLE TIME OF THE METHODS FOR WELDING ON A DOOR (UNIT: SECONDS)

	Scenario 1		Scenario 2	
	Average	Worst	Average	Worst
Conventional	**5±0**	**5**	7±0	**7**
SCIP	**5±0**	**5**	N.S	N.S
BARON	**5±0**	**5**	N.S	N.S
WGA	**5±0**	**5**	7±0	**7**
LGA	**5±0**	**5**	7±0	**7**
WTS	5.6±0.54	6	9.0±0.70	10
LTS	**5±0**	**5**	8.2±0.45	8
WVNS	5.4±0.55	6	9.4±0.55	10
LVNS	**5±0**	**5**	8.8±0.84	9

TABLE VIII
ENERGY CONSUMPTION OF THE METHODS FOR WELDING ON A DOOR (UNIT: KJ)

	Scenario 1		Scenario 2	
	Average	Worst	Average	Worst
Conventional	4.08±0.60	4.82	11.09±0.38	11.60
SCIP	**3.22±0**	**3.22**	N.S	N.S
BARON	**3.22±0**	**3.22**	N.S	N.S
WGA	**3.22±0**	**3.22**	7.22±0.26	7.50
LGA	**3.22±0**	**3.22**	**7.05±0.23**	**7.48**
WTS	3.79±0.32	4.44	8.33±0.38	8.89
LTS	3.61±0.51	4.44	8.09±0.29	8.40
WVNS	3.55±0.41	4.11	8.21±0.24	8.67
LVNS	3.46±0.38	4.11	7.80±0.54	8.91

B. Welding on a Door

Tables VII and VIII compare the cycle time and energy consumption for the considered scenarios of welding on a door (Scenarios 1-2). These two tables show that all the studied methods always obtain solutions for Scenario 1. Differently, SCIP and BARON cannot always find feasible solutions for Scenario 2, which schedules only ten tasks for two robots. These results indicate that the formulated MINLP problem (even for a small-scale scenario) suffers from computational intractability when commercial solvers are used.

Table VII shows that, for every experiment of Scenarios 1 and 2, both WGA and LGA compute the shortest cycle time, which is the same achieved by the conventional method (when minimizing the cycle time only). As a result, the average values of WGA, LGA, and the conventional are equal to their worst values, both for Scenarios 1 and 2.

Table VIII indicates that the average and worst results of the proposed WGA achieve the lowest energy consumption. For Scenario 1, WGA and LGA compute the minimum energy consumption as the one determined by SCIP and BARON. The energy reduction of these four methods for Scenario 1 is about 21%. For Scenario 2, when using WGA and LGA, the energy

TABLE IX
AVERAGE COMPUTATION TIME OF WELDING ON A DOOR (UNIT:SECONDS)

	Scenario 1	Scenario 2
Conventional	11.47±0.38	19.51±0.42
SCIP	35.66±0.92	600±0
BARON	600±0	600±0
WGA	14.87±0.06	22.14±0.05
LGA	26.04±0.10	38.23±0.19
WTS	14.98±0.37	23.58±0.35
LTS	27.60±0.30	40.86±0.68
WVNS	15.17±0.40	24.21±0.80
LVNS	26.93±0.35	39.25 ±0.75

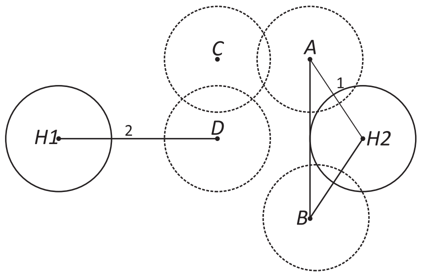


Fig. 5. Geometrical overview of Scenario 1.

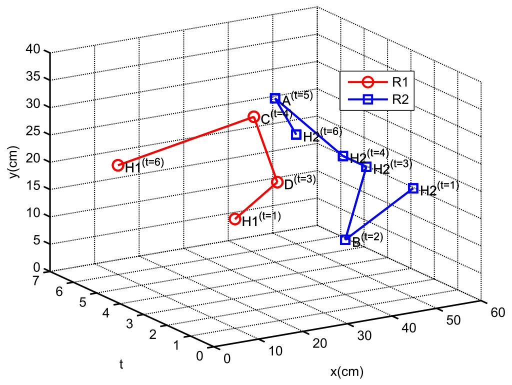


Fig. 6. Planned routes via the conventional method for Scenario 1.

is decreased by around 35% in comparison to the conventional method. In summary, for these two scenarios, the proposed methods efficiently optimize the energy consumption of robot routes without deteriorating the cycle time.

Table IX records the average computation time of different methods. The SCIP solver finds the optimal solution in a short computation time for Scenario 1 while it fails to find a solution for Scenario 2 within the maximum computation time. The solver BARON cannot provide a solution for Scenarios 1-2. The computation times of the studied metaheuristics (WGA, LGA, WTS, LTS, WVNS, and LVNS) compared with the one of the conventional method.

We next present the planned routes of Scenario 1 obtained by the conventional method and WGA. The geometrical overview is given in Fig. 5. H_1 and H_2 are the home nodes for R_1 and R_2. A, B, C, and D are task nodes. Nodes A and B are assigned to R_2, while nodes C and D are assigned to R_1. The parameters P_{ij}^1–P_{ij}^5, which are computed by following the formula (8) in [22].

Fig. 6 shows the detailed routes of each robot obtained by the conventional method including the fixed motion times between tasks nodes. As given in Fig. 6, the computed node sequences for each

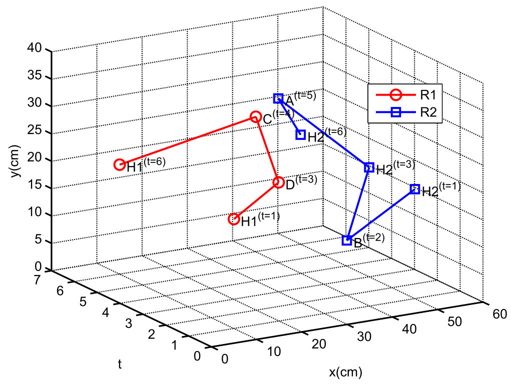


Fig. 7. Planned routes via the proposed WGA for Scenario 1.

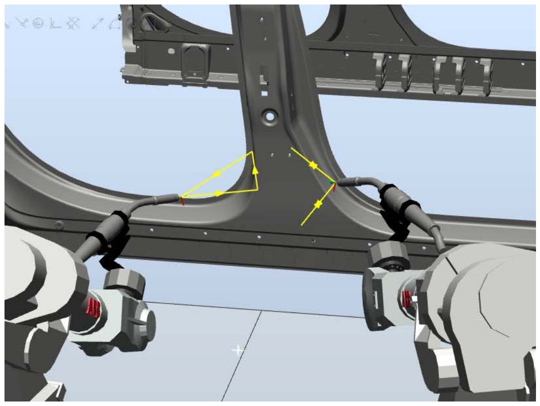


Fig. 8. The planned routes via the proposed WGA for Scenario 1.

robot are as follows: R_1: $H_1 \rightarrow D \rightarrow C \rightarrow H_1$; R_2: $H_2 \rightarrow B \rightarrow H_2 \rightarrow H_2 \rightarrow A \rightarrow H_2$. To avoid the collision with the robot R_1, the robot R_2 returns from node B to home position H_2 and waits at node H_2 between $t = 3$ and $t = 4$ to avoid collisions.

Fig. 7 details the obtained routes by solving the MINLP problem via WGA. The computed visited node sequences are as follows: R_1: $H_1 \rightarrow D \rightarrow C \rightarrow H_1$; R_2: $H_2 \rightarrow B \rightarrow H_2 \rightarrow A \rightarrow H_2$. When using the proposed FTSN model, R_2 can increase the motion time between H_2 and A. The kinetic energy can thus be reduced from waiting at node H_2 as shown in Fig. 6. Meanwhile, the collision between R_1 and R_2 is avoided, as shown both in Fig. 6 and Fig. 7.

Fig. 8 and 9 show the planned and simulated routes obtained by WGA and implemented in RobotStudio for Scenario 1. The planned and executed routes are marked in yellow and blue, respectively. The geometry of the included nodes is consistent with Fig. 5. It can be seen from Fig. 8 and 9 that the executed routes of R_1 and R_2 are the same as the planned routes when using WGA. However, there is a small delay between the planned schedule and the executed schedule, as given in Fig. 10. The reason could be the physical constraints of the robot (e.g., limited joint torques), which are neglected in our robot scheduling problem.

C. Welding on an Underbody

Table X, Table XI, and Table XII record the performance of the studied methods for Scenarios 3-7 to weld on the automotive underbody.

TABLE X

CYCLE TIME OF THE METHODS FOR WELDING ON AN UNDERBODY (UNIT: SECONDS)

	Scenario 3		Scenario 4		Scenario 5		Scenario 6		Scenario 7	
	Average	Worst	Average	Worst	Average	Worst	Average	Worst	Average	Worst
SCIP	N.S	N.S	N.S	N.S	N.S	N.S	N.S	N.S	N.S	N.S
BARON	N.S	N.S	N.S	N.S	N.S	N.S	N.S	N.S	N.S	N.S
Conventional	**18.00±0**	**18**	**25.90±0.88**	**27**	**17.20±0.42**	**18**	**25.50±0.53**	**26**	**32.20±0.42**	**33**
WGA	18.20±0.42	19	26.70±0.67	**27**	17.60±0.70	19	26.10±0.74	27	32.40±0.52	**33**
LGA	**18.00±0**	**18**	**25.90±0.88**	**27**	**17.20±0.42**	**18**	**25.50±0.53**	**26**	**32.20±0.42**	**33**
WTS	21.20±0.79	22	29.50±0.71	30	21.60±1.42	23	28.70±1.06	30	37.40±0.97	38
LTS	20.50±0.53	21	29.00±0.82	30	20.80±1.40	23	28.50±1.08	30	36.40±0.97	38
WVNS	20.80±0.42	21	29.70±1.25	31	22.40±1.46	24	29.40±1.17	31	38.20±1.14	38
LVNS	20.60±0.52	21	29.40±1.51	31	21.70±1.25	24	29.10±1.10	31	37.70±1.25	38

TABLE XI

ENERGY CONSUMPTION OF THE METHODS FOR WELDING ON AN UNDERBODY (UNIT:KJ)

	Scenario 3		Scenario 4		Scenario 5		Scenario 6		Scenario 7	
	Average	Worst	Average	Worst	Average	Worst	Average	Worst	Average	Worst
SCIP	N.S	N.S	N.S	N.S	N.S	N.S	N.S	N.S	N.S	N.S
BARON	N.S	N.S	N.S	N.S	N.S	N.S	N.S	N.S	N.S	N.S
Conventional	14.01±0.76	14.90	16.76±0.58	17.45	14.56±0.98	15.71	24.42±1.35	26.03	30.85±2.52	33.85
WGA	8.63±0.32	8.98	11.38±0.35	11.82	9.34±0.49	9.92	16.08±0.77	**16.91**	20.13±1.12	**20.97**
LGA	**8.54±0.27**	**8.96**	**11.35±0.27**	**11.65**	**9.15±0.37**	**9.78**	**15.83±1.03**	16.94	**19.81±0.99**	20.98
WTS	9.43±0.21	9.68	14.48±0.48	15.10	10.08±0.54	10.90	20.54±0.27	20.99	23.76±0.24	23.96
LTS	9.34±0.21	9.63	14.03±0.43	11.47	9.28±0.21	10.03	19.44±0.24	19.94	22.63±1.10	23.93
WVNS	9.18±0.46	9.73	14.51±0.41	15.31	10.29±0.56	11.24	19.72±1.17	20.77	23.21±0.95	23.92
LVNS	9.02±0.39	9.56	14.04±0.34	14.54	9.80±0.27	10.23	19.49±0.96	20.68	22.85±1.14	23.58

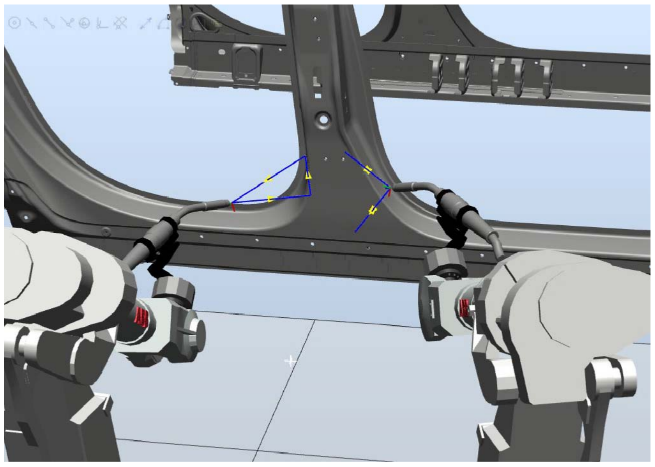


Fig. 9. The simulated routes implemented in RobotStudio for Scenario 1.

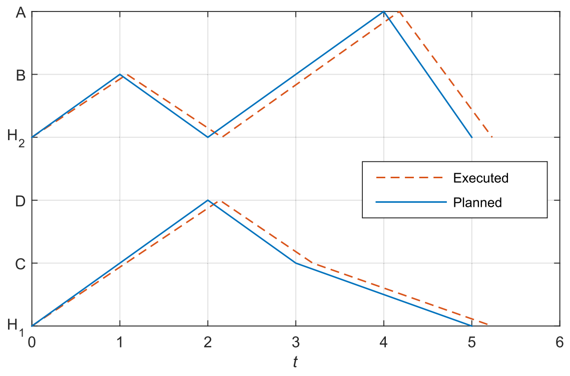


Fig. 10. Planned and simulated task schedules for Scenario 1.

Table X presents the average and worst values of the cycle time for the considered methods. Both the solvers SCIP and BARON cannot obtain any solution for all the scenarios of welding on the underbody, as the numbers of tasks and robots

TABLE XII

AVERAGED COMPUTATIONAL TIMES REGARDING WELDING ON AN UNDERBODY (UNIT:SECONDS)

	Scenario 3	Scenario 4	Scenario 5	Scenario 6	Scenario 7
SCIP	3600±0	3600±0	3600±0	3600±0	3600±0
BARON	3600±0	3600±0	3600±0	3600±0	3600±0
Conventional	23.28±0.42	33.44±0.21	69.76±1.01	70.86±0.31	79.83±0.50
WGA	27.42±1.37	34.37±1.28	73.07±0.62	78.12±0.15	83.36±0.75
LGA	49.89±1.27	67.90±0.76	143.89±1.84	148.66±1.80	164.86±0.48
WTS	29.15±0.85	37.20±0.82	76.37±1.29	80.26±1.09	85.26±1.45
LTS	54.69±0.34	72.83±0.27	148.06±0.2	152.80±0.27	166.97±0.92
WVNS	29.67±0.74	36.34±1.04	75.96±1.20	80.33±0.54	86.88±0.77
LVNS	55.25±0.29	71.85±0.26	147.54±1.25	152.76±0.20	168.43±0.24

regarding these scenarios are more complex than the ones of welding on the door. Regarding the cycle time, LGA and the conventional method achieve the minimal value, both for the average and the worst experiment results. This is because LGA uses the minimal value of the cycle time as a constraint when searching for the most energy-efficient solution. Although WGA performs better than WTS, LTS, WVNS, and LVNS methods, the weighted-sum strategy is not as good as the lexicographic strategy when the cycle time is prioritized.

Table XI records the average and worst values of the energy consumption for Scenarios 3-7. Both LGA and WGA reduce the energy consumption when compared to the conventional method. Regarding the average values, LGA obtains the lowest value for all these scenarios, and its energy reduction is at least 30% when compared to the conventional method. As for the worst case, although LGA is slightly worse than the WGA algorithm for Scenarios 6 and 7, their difference is very small. LGA is still the most efficient searching for the lowest energy consumption while minimizing the shortest cycle time.

Table XII presents the averaged computation time of LGA and WGA in comparison to the other methods for Scenarios 3-7. Table XII confirms that SCIP and BARON fail to find a feasible solution. The conventional method, and WGA and LGA efficiently solve all the scenarios in a reasonable computation time. Since LGA needs first to compute the

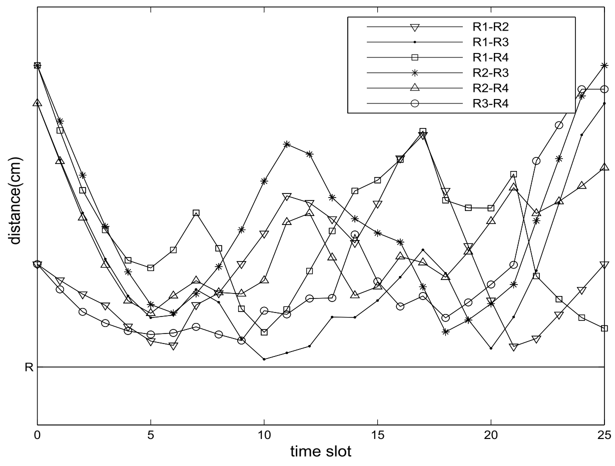


Fig. 11. Relative distances between each pair of robots via the FTSN model for Scenario 6.

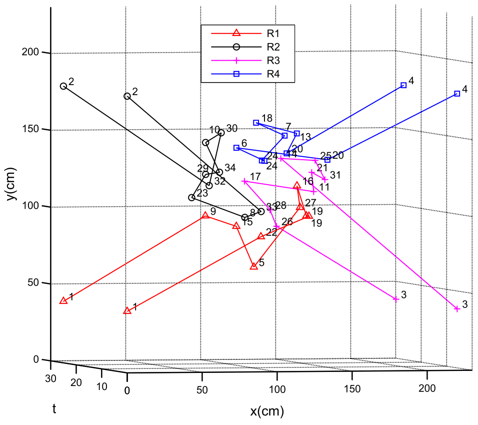


Fig. 12. Robot routes for Scenario 6 based on the FTSN model.

minimal cycle time, the total computation time takes longer than WGA.

We next discuss the planned routes based on the FTSN model and WGA. Fig. 11 gives the relative distance of any two robots based on the FTSN model for Scenario 6. Fig. 11 shows that the FTSN model avoids collisions between robots for each time slot. Fig. 12 illustrates the corresponding routes. All the robots start from their home nodes and ultimately return to these home nodes. Fig. 13 shows their TCP traces in the welding process. These robots move in a collision-free and energy-aware way among the predefined tasks, and the traces are demonstrated in RobotStudio.

D. Relationship Between These Two Objectives

This part discusses the trade-off between the cycle time and energy consumption minimization for the studied energy-efficient routing problem.

In general, our paper aims to achieve high productivity with low energy consumption. The energy reduction is thus expected to be optimized when the shortest cycle time is considered. For energy reduction purposes, the solution set includes Pareto optimal solutions (also called non-dominated solutions) and weakly Pareto optimal solutions [41]. We use the coverage area of these two types of solutions as the metric to evaluate the effects of energy reduction. The coverage

TABLE XIII
SIZES OF EACH COVERAGE AREAS

Scenario	The ε-constraint	NSGA-II
1	15.1486	2.3726
2	22.7829	19.4616
3	33.4765	10.3363
5	44.5000	0.9177
6	184.8293	6.1283

TABLE XIV
COMPUTED PARETO SOLUTIONS BY THE ε-CONSTRAINT METHOD FOR SCENARIO 5

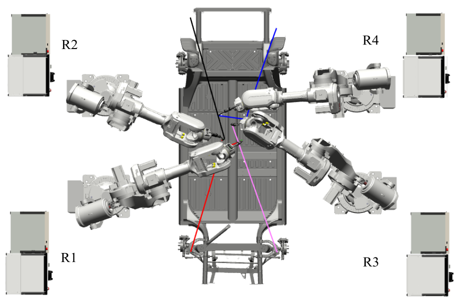
Iterations	Computation times (Sec)	F_c (Sec)	F_e (kJ)
1	73.31	18	15.75
2	74.94	18	14.97
3	74.91	18	11.51
4	73.97	18	9.37
5	74.60	19	8.07
6	73.28	19	6.87
7	73.84	21	6.74
8	74.83	21	5.91
9	74.58	23	5.18

area metric is the 2-D form of the hypervolume indicator, which is a commonly-used metric to evaluate the dominance in multi-objective optimization [42], [43]. We compare ε-constraint method with Non-dominated Sorting Genetic Algorithm II (NSGA-II), which is a state-of-the-art algorithm used in multi-objective optimization [44].

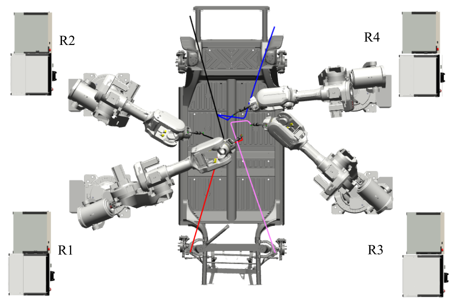
Fig. 14 shows the results obtained by the ε-constraint method (presented in Algorithm 2) and NSGA-II for the selected scenarios. In Fig. 14, for each scenario, the energy consumption of multiple robots is reduced considerably for the shortest cycle time. Table XIII, which compares the coverages areas of the two methods, shows that the proposed ε-constraint has a larger coverage area. The ε-constraint method computes the set of weakly Pareto optimal solutions and Pareto optimal solutions. The weakly Pareto optimal solutions allow us to see the changes in energy reduction for the minimum cycle time. NSGA-II focuses on finding non-dominated points in the Pareto set, and the changes of energy reduction for the minimum cycle time cannot be easily detected.

The extreme point with the shortest cycle time in the Pareto set of these two methods can be compared in Fig. 14. This extreme point is defined as the point with minimal energy consumption and the shortest cycle time, and this point can be useful for the manufacturer to achieve high productivity with low energy consumption. Comparing the extreme points with the shortest cycle time, in general, the ε-constraint method has a shorter cycle time than the NSGA-II method. This indicates that the NSGA-II method may have difficulties finding good extreme points, due to its limited exploration ability for MINLPs, as observed in [45].

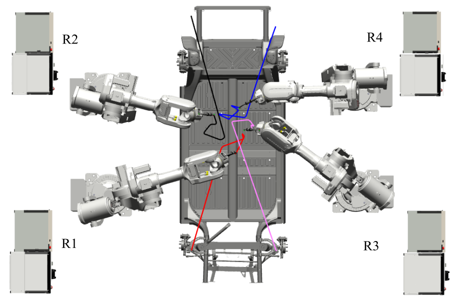
We then discuss the solution obtained by the ε-constraint method for Scenario 5 as an illustrative example. In Fig. 14(d), when obtaining the minimum cycle time (18 seconds), the energy consumption of multiple robots can be further reduced (from about 16 kJ to about 9 kJ). It is thus possible to minimize energy consumption for the minimized cycle time. Fig. 14(d) also shows that, when the energy consumption is lower than its value for the minimum cycle time, the cycle time increases accordingly.



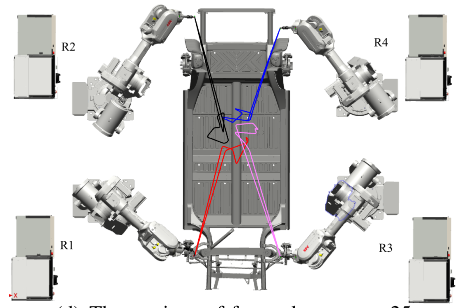
(a) The motions of four robots at $t = 5$.



(b) The motions of four robots at $t = 9$.

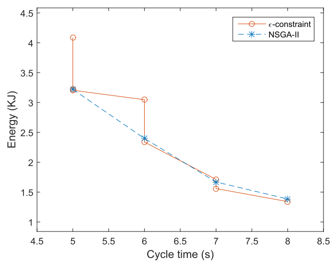


(c) The motions of four robots at $t = 12$.

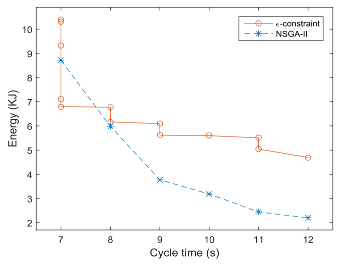


(d) The motions of four robots at $t = 25$.

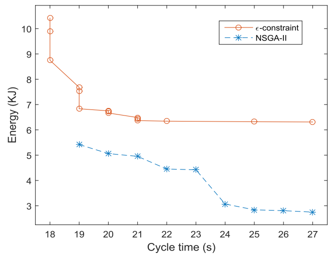
Fig. 13. Illustration of robot traces using the FTSN model for Scenario 6.



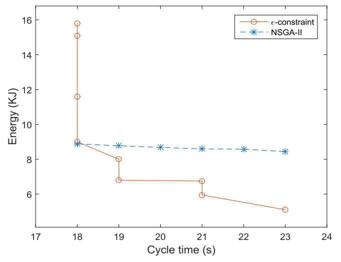
(a) Scenario 1.



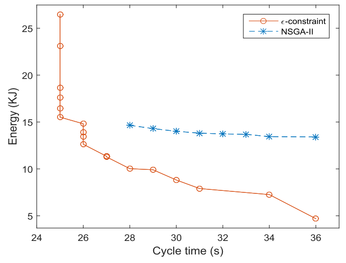
(b) Scenario 2.



(c) Scenario 3.



(d) Scenario 5.



(e) Scenario 6.

Fig. 14. Pareto frontiers computed via the ε-constraint method for the selected scenarios.

Table XIV shows that the ε-constraint method computes four non-dominated solutions (at iterations 4, 6, 8 and 9). For the minimal cycle time ($F_c = 18$), energy consumption can be reduced considerably (from 15.75 KJ to 9.37 KJ) when compared with the case in which F_e is not considered. The solution at iteration 4 becomes an extreme point of the Pareto frontier regarding the minimum cycle time. Since Algorithm 2 is based on GA, the computation times at different iterations are quite close to each other, considering that at each iteration a single objective optimization problem is solved by the same GA configuration.

V. Conclusion and Future Research

This paper studies the energy-efficient robot routing problem for a multi-robot station in manufacturing cells. Our paper minimizes the cycle time and energy consumption jointly and meets the expectation of the producers to save energy for the shortest cycle time. We propose a flexible time-space network model and a customized GA to enable energy-aware and collision-free routing of the robots. Lexicographic and weighted-sum strategies are considered to minimize energy consumption while considering the minimal cycle time. Moreover, we present an ε-constraint algorithm to study the trade-off of the two objectives for the considered MINLPs.

From the numerical results, optimizing task sequences and motion times jointly reduces the energy consumption considerably (up to 30%) without changing the quality of cycle time. The lexicographic formulation performs slightly better than the weighted-sum formulation. Since the decision variables (task orders and motion times) of the studied objectives are closely correlated, the weighted-sum formulation is less computationally efficient than the lexicographic one. The results of the ε-constraint method confirm the potential of energy consumption reduction for the shortest cycle time. The ε-constraint method computes the set of weakly Pareto optimal solutions and Pareto optimal solutions, allowing us to see the changes in energy reduction for the minimum cycle time.

Future research will extend from the robot routing problem of planar motions to 3-D moves. This extended problem is more complicated and needs to consider the movements of multi-joints.

APPENDIX

The initialization of the solution X is given in Algorithm 3.

Algorithm 3 Initialize a Solution X

```
1:  for $k = 1 : m$ do
2:      generate a sequence $s_k$ of all elements from $V_k$ randomly
3:      if $t_k > V_k$ then
4:          for $l = V_k + 1$ to $t_k$ do
5:              select an element from $V_k$ randomly for $s_k(l)$
6:          end for
7:      end if
8:      swap the position of the first $h_k$ and the last task in $s_k$
9:      let $S_{k,1}(1) = h_k,\ S_{k,1}(2 : t_k) = s_k(1 : t_k - 1)$
10:     let $S_{k,2}(1 : t_k - 1) = s_k(1 : t_k - 1),\ S_{k,1}(t_k) = h_k$
11:     Compose $M_k$ by letting $t_{ijk} = t_{ij}^{\min}$
12:     Compose $X_{s,k}$ by based on $S_{k,1}$, $S_{k,2}$ and $t_{ijk}$
13: end for
14: Compose $X_s$ and $X_t$ based on $X_{s,k}$ and $M_k$, respectively
```

REFERENCES

[1] J. Li, H. Sang, Q. Pan, P. Duan, and K. Gao, "Solving multi-area environmental/ economic dispatch by Pareto-based chemical-reaction optimization algorithm," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 5, pp. 1240–1250, Sep. 2019.

[2] Z. Zhao, M. Zhou, and S. Liu, "Iterated greedy algorithms for flow-shop scheduling problems: A tutorial," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 3, pp. 1941–1959, Jul. 2022.

[3] T. C. Lopes, C. G. S. Sikora, R. G. Molina, D. Schibelbain, L. C. A. Rodrigues, and L. Magatão, "Balancing a robotic spot welding manufacturing line: An industrial case study," *Eur. J. Oper. Res.*, vol. 263, no. 3, pp. 1033–1048, Dec. 2017.

[4] D. Spensieri, J. S. Carlson, F. Ekstedt, and R. Bohlin, "An iterative approach for collision free routing and scheduling in multirobot stations," *IEEE Trans. Autom. Sci. Eng.*, vol. 13, no. 2, pp. 950–962, Apr. 2016.

[5] X. Wang, X. Zhou, Z. Xia, and X. Gu, "A survey of welding robot intelligent path optimization," *J. Manuf. Processes*, vol. 63, pp. 14–23, Mar. 2021.

[6] X. Wang, Z. Xia, X. Zhou, Y. Guo, X. Gu, and H. Yan, "Multiobjective path optimization for arc welding robot based on DMOEA/D-ET algorithm and proxy model," *IEEE Trans. Instrum. Meas.*, vol. 70, pp. 1–13, 2021.

[7] X. Wang, J. Wei, X. Zhou, Z. Xia, and X. Gu, "Dual-objective collision-free path optimization of arc welding robot," *IEEE Robot. Autom. Lett.*, vol. 6, no. 4, pp. 6353–6360, Oct. 2021.

[8] L. Bukata, P. Šůcha, Z. Hanzalek, and P. Burget, "Energy optimization of robotic cells," *IEEE Trans. Ind. Informat.*, vol. 13, no. 1, pp. 92–102, Feb. 2017.

[9] M. Brossog *et al.*, "Reducing the energy consumption of industrial robots in manufacturing systems," *Int. J. Adv. Manuf. Technol.*, vol. 78, no. 5, pp. 1315–1328, 2015.

[10] G. A. Korsah, A. Stentz, and M. B. Dias, "A comprehensive taxonomy for multi-robot task allocation," *Int. J. Robot. Res.*, vol. 32, no. 12, pp. 1495–1512, 2013.

[11] E. Åblad, D. Spensieri, R. Bohlin, and J. S. Carlson, "Intersection-free geometrical partitioning of multirobot stations for cycle time optimization," *IEEE Trans. Autom. Sci. Eng.*, vol. 15, no. 2, pp. 842–851, Oct. 2017.

[12] I. Suemitsu, K. Izui, T. Yamada, S. Nishiwaki, A. Noda, and T. Nagatani, "Simultaneous optimization of layout and task schedule for robotic cellular manufacturing systems," *Comput. Ind. Eng.*, vol. 102, pp. 396–407, Dec. 2016.

[13] L. Jin and S. Li, "Distributed task allocation of multiple robots: A control perspective," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 48, no. 5, pp. 693–701, May 2018.

[14] J. Xin, C. Meng, F. Schulte, J. Peng, Y. Liu, and R. Negenborn, "A time-space network model for collision-free routing of planar motions in a multi-robot station," *IEEE Trans. Ind. Informat*, vol. 16, no. 10, pp. 6413–6422, Oct. 2020.

[15] S. Gürel, H. Gultekin, and V. E. Akhlaghi, "Energy conscious scheduling of a material handling robot in a manufacturing cell," *Robot. Comput.-Integr. Manuf.*, vol. 58, pp. 97–108, Aug. 2019.

[16] M. Gadaleta, M. Pellicciari, and G. Berselli, "Optimization of the energy consumption of industrial robots for automatic code generation," *Robot. Comput.-Integr. Manuf.*, vol. 57, pp. 452–464, Jun. 2019.

[17] L. Bukata, P. Šůcha, and Z. Hanzálek, "Optimizing energy consumption of robotic cells by a Branch & Bound algorithm," *Comput. Oper. Res.*, vol. 102, pp. 52–66, Feb. 2019.

[18] E. Glorieux, S. Riazi, and B. Lennartson, "Productivity/energy optimisation of trajectories and coordination for cyclic multi-robot systems," *Robot. Comput.-Integr. Manuf.*, vol. 49, pp. 152–161, Feb. 2018.

[19] M. H. F. Zarandi, H. Mosadegh, and M. Fattahi, "Two-machine robotic cell scheduling problem with sequence-dependent setup times," *Comput. Oper. Res.*, vol. 40, no. 5, pp. 1420–1434, May 2013.

[20] B. Vaisi, "A review of optimization models and applications in robotic manufacturing systems: Industry 4.0 and beyond," *Decis. Anal. J.*, vol. 2, Mar. 2022, Art. no. 100031.

[21] J. Xin, R. R. Negenborn, F. Corman, and G. Lodewijks, "Control of interacting machines in automated container terminals using a sequential planning approach for collision avoidance," *Transp. Res. C, Emerg. Technol.*, vol. 60, pp. 377–396, Nov. 2015.

[22] A. Vergnano *et al.*, "Modeling and optimization of energy consumption in cooperative multi-robot systems," *IEEE Trans. Autom. Sci. Eng.*, vol. 9, no. 2, pp. 423–428, Apr. 2012.

[23] S. Burer and A. N. Letchford, "Non-convex mixed-integer nonlinear programming: A survey," *Surv. Oper. Res. Manage. Sci.*, vol. 17, no. 2, pp. 97–106, 2012.

[24] M. Ghahramani, Y. Qiao, M. Zhou, A. O. Hagan, and J. Sweeney, "AI-based modeling and data-driven evaluation for smart manufacturing processes," *IEEE/CAA J. Autom. Sinica*, vol. 7, no. 4, pp. 1026–1037, Jul. 2020.

[25] Y. Hou, N. Wu, M. Zhou, and Z. Li, "Pareto-optimization for scheduling of crude oil operations in refinery via genetic algorithm," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 47, no. 3, pp. 517–530, Mar. 2017.

[26] H. Yuan, J. Bi, and M. Zhou, "Profit-sensitive spatial scheduling of multi-application tasks in distributed green clouds," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 3, pp. 1097–1106, Jul. 2020.

[27] L. Tang, A. D'Ariano, X. Xu, Y. Li, X. Ding, and M. Samà, "Scheduling local and express trains in suburban rail transit lines: Mixed–integer nonlinear programming and adaptive genetic algorithm," *Comput. Oper. Res.*, vol. 135, Nov. 2021, Art. no. 105436.

[28] J. Tang, G. Liu, and Q. Pan, "A review on representative swarm intelligence algorithms for solving optimization problems: Applications and trends," *IEEE/CAA J. Autom. Sinica*, vol. 8, no. 10, pp. 1627–1643, Oct. 2021.

[29] J. Xin, B. Yu, A. D'Ariano, H. Wang, and M. Wang, "Time-dependent rural postman problem: Time-space network formulation and genetic algorithm," *Oper. Res.*, vol. 22, no. 3, pp. 2943–2972, 2021.

[30] M. Gendreau *et al.*, *Handbook of Metaheuristics*, vol. 2. New York, NY, USA: Springer, 2010.

[31] A. H. Halim and I. Ismail, "Combinatorial optimization: Comparison of heuristic algorithms in travelling salesman problem," *Arch. Comput. Methods Eng.*, vol. 26, no. 2, pp. 367–380, 2019.

[32] P. Wu, A. Che, F. Chu, and M. Zhou, "An improved exact ε-constraint and cut-and-solve combined method for biobjective robust lane reservation," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 3, pp. 1479–1492, Jun. 2015.

[33] A. D'Ariano, L. Meng, G. Centulio, and F. Corman, "Integrated stochastic optimization approaches for tactical scheduling of trains and railway infrastructure maintenance," *Comput. Ind. Eng.*, vol. 127, pp. 1315–1335, Jan. 2019.

[34] N. V. Sahinidis, "BARON 17.8.9: Global optimization of mixed-integer nonlinear programs," User's Manual, Tech. Rep., 2017.

[35] A. Heidari, Z. Y. Dong, D. Zhang, P. Siano, and J. Aghaei, "Mixed-integer nonlinear programming formulation for distribution networks reliability optimization," *IEEE Trans. Ind. Informat.*, vol. 14, no. 5, pp. 1952–1961, May 2018.

[36] S. Vigerske and A. Gleixner, "SCIP: Global optimization of mixed-integer nonlinear programs in a branch-and-cut framework," *Optim. Methods Softw.*, vol. 33, no. 3, pp. 563–593, 2018.

[37] M. R. Bussieck and S. Vigerske, "MINLP solver software," in *Wiley Encyclopedia of Operations Research and Management Science*. Hoboken, NJ, USA: Wiley, 2011, pp. 1–17.

[38] F. Glover and E. Taillard, "A user's guide to Tabu search," *Ann. Oper. Res.*, vol. 41, no. 1, pp. 1–28, 1993.

[39] P. Hansen, N. Mladenović, R. Todosijević, and S. Hanafi, "Variable neighborhood search: Basics and variants," *EURO J. Comput. Optim.*, vol. 5, no. 3, pp. 423–454, Sep. 2017.

[40] ABB. *Robotstudio: The World's Most Used Offline Programming Tool for Robotics*. Accessed: Jun. 5, 2022. [Online]. Available: https://new.abb.com/products/robotics/robotstudio

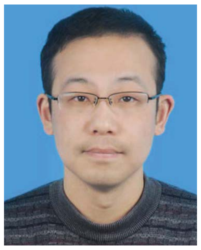
[41] M. Elmusrati, H. El-Sallabi, and H. Koivo, "Applications of multi-objective optimization techniques in radio resource scheduling of cellular communication systems," *IEEE Trans. Wireless Commun.*, vol. 7, no. 1, pp. 343–353, Jan. 2008.

[42] F. Corman, A. D'Ariano, D. Pacciarelli, and M. Pranzo, "Bi-objective conflict detection and resolution in railway traffic management," *Transp. Res. C, Emerg. Technol.*, vol. 20, no. 1, pp. 79–94, 2012.

[43] D. Di Nucci, A. Panichella, A. Zaidman, and A. De Lucia, "A test case prioritization genetic algorithm guided by the hypervolume indicator," *IEEE Trans. Softw. Eng.*, vol. 46, no. 6, pp. 674–696, Jun. 2020.

[44] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Aug. 2002.

[45] A. Jaber, P. Lafon, and R. Younes, "A branch-and-bound algorithm based on NSGAII for multi-objective mixed integer nonlinear optimization problems," *Eng. Optim.*, vol. 54, no. 6, pp. 1–19, 2021.



Jianbin Xin (Member, IEEE) received the B.Sc. degree in electrical engineering from Xidian University, Xi'an, China, in 2007, the M.Sc. degree in control science and engineering from Xi'an Jiaotong University, Xi'an, in 2010, and the Ph.D. degree in operational control for logistics from the Delft University of Technology, Delft, The Netherlands, in 2015.

Currently, he is an Associate Professor with the School of Electrical and Information Engineering, Zhengzhou University, China. His research interests include planning and control of smart logistics systems and cooperative robots.

Chuang Meng received the bachelor's degree in electrical engineering from the Luoyang Institute of Science and Technology in 2018. He is currently pursuing the master's degree with the School of Electrical Engineering, Zhengzhou University. His research interest includes planning of multiple robots for manufacturing and logistics.

Andrea D'Ariano received the B.S. and M.S. degrees in computer science, automation, and management engineering from Roma Tre University and the Ph.D. degree from the Department of Transport and Planning, Delft University of Technology, The Netherlands, in April 2008, under the supervision of Prof. I. A. Hansen.

Currently, he is an Associate Professor with the Department of Engineering, Roma Tre University. He is an Associate Editor of well-known international journals such as *Transportation Research Part—B: Methodological*, *Transportation Research Part—C: Emerging Technologies*, and *Transportation Research Part—E: Logistics and Transportation Review*; and conferences such as IEEE ITSC. His main research interests include the study of scheduling and routing problems with application to transportation and logistics.

Frederik Schulte received the Diploma degree in industrial engineering with majors in information systems and logistics from the Technical University of Hamburg and the Ph.D. degree in logistics from the University of Hamburg.

Currently, he is an Assistant Professor with the Department of Maritime Transport and Technology, Delft University of Technology. His research focuses on collaborative transportation and implications of the fourth industrial revolution.

Jinzhu Peng (Member, IEEE) received the B.E., M.E. and Ph.D. degrees from Hunan University, China, in 2002, 2005, and 2008, respectively.

From 2009 to 2011, he worked as a Post-Doctoral Fellow with the University of New Brunswick, Canada. Currently, he is a Professor with the School of Electrical and Information Engineering, Zhengzhou University, China. His research interests include robotic control, compliant control, and human–robot interactions and collaborations.

Rudy R. Negenborn received the M.Sc. degree in computer science from Utrecht University in 2003 and the Ph.D. degree from the Delft Center for Systems and Control, Delft University of Technology, in 2007.

He is currently a Full Professor of Multi-Machine Operations and Logistics and the Head of the Transport Engineering and Logistics Section, Delft University of Technology. His fundamental research interests are in the areas of distributed control, multi-agent systems, model predictive control, and optimization.