



Simulation-based metaheuristic optimization algorithm for material handling

Carolina Saavedra Sueldo^{1,2} · Ivo Perez Colo^{1,2} · Mariano De Paula^{1,2} · Sebastián A. Villar^{1,2} · Gerardo G. Acosta^{1,2}

Received: 1 September 2023 / Accepted: 9 January 2024

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2024

Abstract

Modern technologies and the emergent Industry 4.0 paradigm have empowered the emergence of flexible production systems suitable to cope with custom product demands, typical in this era of competitive marketplaces. However, production flexibility claims periodic changes in the setup of production facilities. The level of flexibility of a production process increases as the reconfiguration capacity of its facilities increases. Nevertheless, doing that efficiently requires accurate coordination between productive resources, task planning, and decision-making systems aiming to maximize value for the client, minimizing non-added-value production tasks, and continuous process improvement. In a manufacturing system, material handling within manufacturing facilities is one of the major non-value-added tasks strongly affected by changes in plant floor layouts and demands for producing customized products. This work proposes a metaheuristic simulation-based optimization methodology to address the material handling problem in dynamic environments. Our proposed approach integrates optimization, discrete event simulation, and artificial intelligence methods. Our proposed optimization algorithm is mainly based on the ideas of the novel population-based optimization algorithm called Q-learning embedded Sine Cosine Algorithm, inspired by the Sine Cosine Algorithm. Unlike those, our proposed approach can deal with discrete optimization problems. It includes in its formulation a reinforcement learning embedded algorithm for the self-learning of the parameters of the metaheuristic optimization algorithm, and discrete event simulation is used for simulating the shop floor operations. The performance of the proposed approach is evaluated through an exhaustive analysis of simple to complex cases. In addition, a comparison is made with other comparable optimization methodologies.

Keywords Optimization · Simulation · Material Handling · Artificial Intelligence · Lean 4.0

I. P. Colo, M. De Paula, S. A. Villar and G. G. Acosta have contributed equally to this work.

✉ Carolina Saavedra Sueldo
carolina.saavedra@fio.unicen.edu.ar

Ivo Perez Colo
ivo.perez@fio.unicen.edu.ar

Mariano De Paula
mariano.depaula@fio.unicen.edu.ar

Sebastián A. Villar
svillar@fio.unicen.edu.ar

Gerardo G. Acosta
ggacosta@fio.unicen.edu.ar

¹ Centro de Investigaciones en Física e Ingeniería del Centro, UNCPBA-CICPBA-CONICET, Olavarría, Buenos Aires, Argentina

Introduction

Today's manufacturing companies are facing a wide range of obstacles and challenges. Production tends to be personalized, and demands are rapidly changing and diversifying. In addition, the speed of change in the environment is increasing. In this context, the paradigm of Industry 4.0 (I4.0) or the Fourth Industrial Revolution refers to a set of productive and institutional transformations. The concept of smart factories appears, where physical objects, machines, products, and even parts are integrated into the company's information net-

² Facultad de Ingeniería, Intelymec, Universidad Nacional del Centro de la Provincia de Buenos Aires (UNCPBA), Olavarría, Buenos Aires, Argentina

works and communicate in real-time horizontally with each other and vertically with customers, users, and suppliers.

Computer systems monitor manufacturing processes, creating a virtual copy of the physical world and making decentralized decisions (Motta et al., 2019). This concept, called Digital Twin (DT), has recently generated much interest in academia and industry (Stączek et al., 2021). Furthermore, this has been accompanied by an increase in related publications. There is excellent potential in operating digital twins as agent-based systems that cooperate towards specific goals, with emerging benefits for the overall system (Jones et al., 2020).

I4.0 also includes technologies such as Cyber-Physical Systems, the Internet of Things, Big Data, Cloud Computing, Cloud Manufacturing, Robotics, and Simulation. Therefore, it considers a broad set of new technologies (software and hardware) that operate in integrated systems (Benitez et al., 2023) intending to achieve greater process flexibility. Moreover, even in these times, there is already talk of Industry 5.0 as a new paradigm, which includes concepts such as a humanized vision of the industry, circular and sustainable manufacturing, and future needs of society (Lu et al., 2023; Barata and Kayser, 2023; Golovianko et al., 2023; Pizoń and Gola, 2023).

On the other hand, Lean Manufacturing is a management philosophy of a company or organization and a long-term strategy (Womack et al., 2007). It was developed in the 1950s in Japan at the Toyota Company and can be summed up with the phrase “do more with less”. In other words, it implies making more efficient use of available resources. Time and material waste are identified and eliminated to maintain quality and reduce manufacturing costs (Alkhorraif et al., 2019). The fundamental part of this philosophy is to focus on the company’s activities that add real value. This concept includes several engineering and management tools and methods, some of which are: 5 S, Just in Time (JIT), Total Productive Maintenance (TPM), Continuous Improvement (Kaizen), Total Quality Management (TQM), Zero Defects, Kanban, Standardization of Tasks, Value Stream Mapping (VSM), among others (Shah and Patel, 2018; Akkari and Valamede, 2020).

The Lean methodology identifies eight wastes of an organization that must be minimized to achieve efficiency and effectiveness in production systems, seven of which were raised by Taiichi Ohno (Ohno, 1988). These wastes are overproduction, waiting, unnecessary movements, transportation, inventory, defects, overprocessing, and unused talent. The minimization of these factors directly impacts production costs and the company’s competitiveness. And identifying these wastes is always the first step.

The ideas of Lean Manufacturing and I4.0 have been integrated into a new organizational paradigm called “Lean 4.0” (Langlotz et al., 2021). Methods from one paradigm can be

supplemented by technology from another, and vice versa. While these paradigms, Lean Manufacturing and I4.0, may seem quite different, Lean 4.0 offers promising opportunities for the future of organizations if they can combine them synergistically to improve efficiency (Marinelli et al., 2021).

Returning to Lean wastes, transportation may be defined as a necessary operation that adds no value to a manufacturing process. Concerning the plant floor, material handling (MH) is vital in a manufacturing facility, but it frequently adds unnecessary time and expense (Adeodu et al., 2023; Yamazaki et al., 2017). The inefficient transportation of raw materials or products in the process directly influences the business’s productivity and profitability. It also impacts other Lean wastes, such as waiting in manufacturing and finished and in-process inventories.

This situation is related to two fundamental problems that have been treated for years in operations optimization: the traveling salesman problem (TSP) and the capacitated vehicle routing problem (CVRP). These problems are considered *NP-hard* combinatorial problems. In the TSP, given a list of nodes and the distance between them, one must find the shortest possible path to visit each node and return to the original one (Wang and Tang, 2021). Formally, an instance of a TSP can be defined in a graph as a set of nodes and edges. Each node has its characteristics, and a possible solution to the problem involves defining a path, that is, a specific sequence of nodes that satisfies the problem’s constraints. On the other hand, in the specific case of a CVRP, there is another node called the warehouse, and the original nodes represent clients, each with a particular demand. Each route starts from the depot and visits a subset of clients sequentially (Yaixin et al., 2022). It is a directed graph. And the goal is to minimize the distance of the tour. This situation is the case with the MH problem on the plant floor, where, given an initial point, different workstations are visited to supply it.

However, in flexible production systems, the shop floor could be modified. Consequently, using graphs-based optimization methods to address lean minimization problems could be cumbersome since they must be refactored every time there is a change in the configuration of the productive system. Instead, using discrete-event simulation-based methods or, even more, digital twin-based strategies gives a more realistic description of the system.

In this context, this study makes a proposal that combines discrete event simulation (DES), optimization, and artificial intelligence (AI) for the MH problem on the plant floor that has been little explored in the literature for flexible workshop floors in manufacturing plants. Our proposed optimization algorithm is mainly based on the ideas of the novel population-based optimization algorithm called Q-learning embedded Sine Cosine Algorithm (QLESCA) (Hamad et al., 2022), inspired by the Sine Cosine Algorithm (SCA) (Mirjalili, 2016) and its discrete version recently presented in

the work of Gupta et al. (2022). In addition, to improve the convergence performance of our proposal, we embed a reinforcement learning algorithm into the formulation. In this way, our proposal can learn a behavioral law of the exploration/exploitation parameter that significantly influences the algorithm's performance.

Notably, during the literature review, a non-significant amount of previous works in SCA applications have been identified for discrete optimization problems and even less so for the MH problem in flexible production systems. In summary, this work's main contributions address minimizing a non-value-added task, mainly focusing on the MH problem where multiple resources are considered, vehicle routing and resource synchronization are posed, and a methodology for solving the proposed formulation is developed.

The paper is structured as follows: in section "Related work", the related work; in section "Methodological background", the methodological background; in section "Meta-heuristic DSCA using RL-optimized parameterization laws for MH optimization", the proposal developed and applied to the case study; in section "Results and discussions", the corresponding results; and finally, in section "Conclusion and future work", the conclusions and future work.

Related work

The concept of Lean 4.0 has emerged strongly in several investigations lately, combining the Lean philosophy with the new paradigm of I4.0 and treating them as complementary elements. Several works can be found where concepts regarding this combination of factors are reviewed or raised, or possible methodologies for their application are specified (Dillinger et al., 2021, 2022; Kolla et al., 2019). Some researchers suggest that a combination of Lean, simulation, and optimization may be crucial in the future to improve organizations' efficiency. This combination can improve the traditional decision-making process, accelerate system improvements and re-configurations, and support organizational learning (Uriarte et al., 2018). However, there are few concrete articles of application in various case studies.

In Cifone et al. (2021), two current forms are proposed for this integration. The first perspective suggests Lean as a foundation for I4.0 implementation, arguing that controlled and optimized processes can be a prerequisite for any digitization process. In contrast, the second perspective shows I4.0 as a necessary complement to Lean, based on the great personalization of the demand that the current situation implies, where I4.0 represents a means that Lean can exploit to adapt to new trends in the world. In this work, we demonstrated this last trend using I4.0 tools for optimizing the MH problem and eliminating Lean waste in a conjunction of both paradigms.

On the other hand, the intersection of advanced technologies and industrial applications implies the development of DT technology and strategies for human-machine collaboration to make more efficient operational environments. In this sense, for the MH task, integrating Autonomous Mobile Robots and robot manipulators (Ghodsian et al., 2022) in the shopfloor could bring new ways for managing the material along the processes of transformation and their manipulation in the warehouses. In the recent work of Stączek et al. (2021), DT technology was used to assess the correctness of the design assumptions adopted for the early phase of implementing an autonomous mobile vehicle in a company's production hall. However, the work mainly focuses on efficiently developing the DT. A step forward is discussed in Kiyokawa et al. (2023), where a review analyzes the prevalent approaches focusing on understanding the current difficulty and complexity definitions and outstanding human-robot collaboration assembly system issues. Similarly, this new trend of human-machine integration is addressed in Lu et al. (2023), where a DT-based framework is presented for coordinating teams of humans and robots, focusing on DT technology as a central communication hub to enable collaboration between these entities. All these advances should allow us to have more precise and real-time information about the operating environment, which means that the case of MH, whether carried out solely by robots or in collaboration with humans, requires efficient optimization algorithms to solve such problems.

Concerning the TSP and the CVRP, it can find several methodologies for solving these problems, such as evolutionary algorithms (Akhand et al., 2015; Zhang and Yang, 2022; Skinderowicz, 2022), various optimization methods (Dündar et al., 2022; Singh et al., 2022), reinforcement learning (RL) (Nazari et al., 2018; Ottoni et al., 2022), and neural networks (Hu et al., 2020; Luo et al., 2022). Within the wide range of optimization, a simple meta-heuristic algorithm called SCA (Mirjalili, 2016) emerged a few years ago, showing excellent results in various types of problems. Currently, it is being used in a plethora of applications and can be found in multiple research in several fields. For example, in Karmouni et al. (2022), it is used to control and improve monitoring performance in a photovoltaic system; in Issa (2021), it contributes to measuring pairs of biological sequences in the context of COVID-19, in Kuo and Wang (2022) it is used in a classification problem with mixed data, in Lyu et al. (2021) it is used for the prediction of the resistance to axial compression of circular columns and in Daoui et al. (2021) it is used for the copyright protection of images.

However, despite the potential of SCA, at least during our literature review, we have not found relevant applications in decision-making systems for the manufacturing process.

The original article shows that the SCA can be highly effective in solving real problems with restricted and

unknown search spaces and that it converges, avoiding local optimum. In addition, it has a reasonable execution time and a straightforward implementation (Benmessaoud et al., 2021). Although this is true, several authors developed improvements or variants of this algorithm that increased its robustness or speed of convergence by adding certain meta-heuristics or hybridizations, such as Abdel-Baset et al. (2019); Li et al. (2018) and Al-qaness et al. (2018).

However, few articles have been found where problems with solutions of a discrete nature are dealt with (Yang et al., 2020), and no works have been developed where SCA can be applied to manufacturing systems or MH problems. We have rescued the article developed by Gupta et al. (2022), where they implement the Discrete Sine Cosine Algorithm (DSCA) and apply it to the programming of urban traffic lights as a basis for our proposal.

On the other hand, to solve routing problems, many works use graph theory for its resolution (Gouveia et al., 2019; Lei et al., 2022; Duhamel et al., 2011). However, using DES and digital models of manufacturing systems, we can not only represent a static layout but also take advantage of the potential of simulators to study complete systems. This allows us to have production statistics, stock, and failures, among others, and get a more realistic representation of the production process for the problem formulation.

In this work, we demonstrate that successful results are obtained without expert knowledge when using the proposed optimization method together with process simulation. This makes it easier for established industries to implement Lean 4.0 ideas, and this concept has been little addressed in the literature for flexible shop floor applications in manufacturing plants.

Methodological background

Discrete event simulation

In the current context, the decision-makers of productive companies are increasingly facing complex situations where they must maximize both production, economic, and sustainability objectives that often conflict with each other. The simulation of systems for the development of virtual models constitutes an essential tool to support decision-making so that companies can evaluate their operational policies to maximize the use of resources and quickly adapt to environmental changes.

Among the simulation methodologies available, DES is mainly used in manufacturing systems to evaluate, test, and improve the performance of processes at low cost, without intervening in them or their actual operation (Furian et al., 2015). DES is the process of encoding the behavior of a complex system as an ordered sequence of well-defined

events. These events occur at particular times and mark a state change in the system (Kiran, 2019). Because no changes occur between events, discrete event models can directly simulate the occurrence time of the next event, and long time intervals can be significantly reduced.

I4.0 has even given way to the concept of digital twins, given the decisive irruption of the industrial Internet of things together with cyber-physical systems, where physical systems are connected with their simulated virtual twins in real-time. This makes monitoring and controlling them efficiently possible (Saavedra Sueldo et al., 2022). According to Semeraro et al. (2021), digital twins are “a set of adaptive models that emulate the behavior of a physical system in a virtual system, obtaining data in real-time to update throughout its life cycle. They replicate the physical system to predict failures and opportunities for change, prescribe actions in real-time to optimize or mitigate unexpected events, and observe and evaluate the operating profile of the system”.

Briefly, discrete event simulation offers the opportunity to test multiple solutions within a short time efficiently. It enables the analysis of various scenarios until the optimal one is discovered. In the context of MH problems, discrete event simulation using a manufacturing plant simulator provides the advantage of working with dynamic environments. It allows for considering perturbations and changes that are challenging to model using traditional methods commonly used in routing problems, such as graphs with vertices and arcs to represent the shop floor. Moreover, DES facilitates comprehensive data collection on industrial systems and different Lean wastes, in addition to evaluating factors like distances, routes, and statistical information across areas like maintenance, production, and quality management. This holistic approach ensures a precise assessment of the impacts of implemented actions on the entire production system (Ben Moussa et al., 2019).

Optimization

Optimization is finding the optimal values of the parameters or variables involved in a problem and maximizing or minimizing the final result. Thus, in an optimization problem formulation, we aim to maximize or minimize an objective function $f(x)$ subject to certain constraints. The objective function can be linear or non-linear. Values that can be controlled and influence the system are called decision variables (Winston, 2004).

There are several approaches to solving different optimization problems (Thirunavukkarasu et al., 2023; Gambella et al., 2021). We can locate stochastic algorithms among the many methods for tackling this sort of issue. These techniques treat optimization problems as black boxes. This means that the derivation of mathematical models is unnecessary because such optimization paradigms simply modify

the system’s inputs and monitor its outputs to maximize or minimize them. Another significant advantage of viewing problems as black boxes is their high adaptability, which means that stochastic algorithms adapt to problems in various domains. In addition, black-box optimization methods are designed to deal with complex systems with complex behavior to be accurately modeled without making excessive simplifications and assumptions that may affect the representation of the system. However, these methods can converge slowly or get stuck in local minima. On the other hand, metaheuristics are fundamental for solving complex optimization problems because they can produce acceptable results in a reasonable amount of time, making them suitable replacements for accurate algorithms (Karimi-Mamaghan et al., 2022).

The stochastic and metaheuristic optimization algorithm used in this article is based on the one developed by Mirjalili (2016) called SCA. This algorithm is simple and effective, and it was proposed to optimize real problems with unknown search spaces. It is a population-based optimization algorithm; that is, it starts with several random initial candidate solutions that are updated when compared with the best available solution. These solutions are close to or far from the best solution based on a mathematical model based on sine and cosine functions. In addition, there are four parameters integrated into the algorithm r_1, r_2, r_3 and r_4 that allow managing the exploration and exploitation of the search space or feasible zone. The fundamental equation is the following:

$$X_i^{t+1} = \begin{cases} X_i^t + r_1 * \sin(r_2) * |r_3 P_i^t - X_i^t|, & r_4 < 0.5 \\ X_i^t + r_1 * \cos(r_2) * |r_3 P_i^t - X_i^t|, & r_4 \geq 0.5 \end{cases} \quad (1)$$

where X_i^t is the position of the current solution at iteration t and dimension i and P_i^t is the position of the best solution found so far at this moment. The formula of r_1 is $r_1 = a - t \frac{a}{T}$, where a is a constant, t the current iteration and T the maximum number of iterations of the algorithm. r_2 is a random number that varies between 0 and 2π , r_3 is a random number that varies between 0 and 2, and r_4 is a random number ranging between 0 and 1. The SCA pseudocode is shown below in Algorithm 1.

Algorithm 1 Sine Cosine Algorithm

Set T = maximum number of iterations
Initialize a set of search agents or solutions (X)
Evaluate each agent by the objective function
Update the best solution obtained so far ($P = X^*$)
While $t < T$:
 Update r_1, r_2, r_3 and r_4
 Update search agents positions using Eq. (1)
 Evaluate each agent by the objective function
 Return the best solution obtained so far as the global solution

The MH problem in a productive system involves finding the sequence of stations that make up a punctual route that minimizes the distance traveled. In this sense, the optimization problem requires dealing with discrete variables. Therefore, it is impossible to use the SCA directly since it is an algorithm for continuous variables. It is decided to adopt a variant called DSCA developed by Gupta et al. (2022). The fundamental equation of the SCA is discretized as follows for the DSCA:

$$X_{i,new}^{t+1} = X_i^t \oplus (C_i^t \otimes (X_d^t \ominus X_i^t)) \quad (2)$$

$$C_i^t = \begin{cases} |r_1 * \sin(r_2)|, & r_3 > 0.5 \\ |r_1 * \cos(r_2)|, & r_3 \leq 0.5 \end{cases} \quad (3)$$

where the parameter $r_1 = \frac{t}{T}$ is linearly ascending between 0 and 1, r_2 is a random number ranging between 0 and 2π , and r_3 is a random number ranging between 0 and 1.

As in the original SCA, X_i^t is the position of the current solution, where X_d^t is the best solution obtained so far. The parameter C_i^t is used to decide the required change amount in the current candidate’s position vector to obtain a new state. That is, in $X_d^t \ominus X_i^t$ the difference of positions of the vectors is calculated: how many components in the solution vector X_i are different from the best solution found X_d . After the multiplication in Eq. 2, the final value is rounded to update the solution, changing the number of characteristics or components obtained. As can be seen by comparing Eq. (1) and (2), in the discrete algorithm, there is one less parameter to consider (r_1, r_2, r_3) concerning SCA. The DSCA pseudocode is shown in Algorithm 2.

Algorithm 2 Discrete Sine Cosine Algorithm

Set T = maximum number of iterations
Initialize a set of search agents or solutions(X)
Evaluate each agent by the objective function
Update the best solution obtained so far (X_d)
While $t < T$:
 Calculate for each agent the difference in characteristics of the vectors: $X_d^t \ominus X_i^t$
 Update r_1, r_2 and r_3 ; and **calculate** C_i^t
 Update search agents positions using Eq. (2)
 Evaluate each agent by the objective function
 Return the best solution obtained so far as the global solution

The performance of the DSCA is highly dependent on the r_i parameters. In this sense, we propose a version with an embedded RL algorithm for the parameters self-tuning policy learning.

Reinforcement learning

Within the different types of existing AI techniques, we can highlight RL as being fundamental to the developments of

recent years. This AI technique involves the interaction of an intelligent agent with its environment, where it learns based on trial and error. Learning from interaction is a fundamental idea underlying almost all theories of learning and intelligence. RL is learning what to do, that is, how to assign situations to actions to maximize a numerical reward signal (Sutton and Barto, 1998).

Formally, the RL problem is described as a Markov decision problem (MDP), which requires the definition of a state vector of the system, the set of possible actions to be taken in each system state, and a reward function that rewards the effects of the actions taken. In this way, the agent generates its knowledge base and adapts its action policy π according to the defined reward function. A policy describes the agent's behavior. In broad terms, it is a mapping from perceived system states to actions to be conducted in those states.

MDPs are a classic formalization of sequential decision-making, where actions influence immediate rewards and subsequent situations or states. In more detail, the agent and the environment interact in a series of discrete time steps $t = 0, 1, 2, 3 \dots$. At each time step t , the agent receives some representation of the state of the system s_t and chooses an action a_t based on that representation. One step forward, due to the taken action, the system evolves to a new state s_{t+1} , and the agent gets a numerical reward r_{w_t} .

One of the RL's problems is the compensation between exploration and exploitation. An RL agent should choose actions aiming to maximize the obtained rewards. However, to discover such actions, the agent must try some actions that it has never tried before. Thus, the agent must exploit what it has already experienced to gain a reward, but it must also explore to make better decisions for future action. Neither exploration nor exploitation can be entirely carried out; the agent must try a variety of acts and gradually improve its behavior. Therefore, it is necessary to consider which hyperparameters are used in the chosen algorithm and how to balance this scenario (Sutton and Barto, 1998).

The *Q-learning* algorithm is one of the best-known RL tabular algorithms. It was developed by Watkins in 1989, and it is an incremental method that works by successively improving the quality of particular actions in particular states (Watkins and Dayan, 1992). The standard *Q-learning* was developed to deal with finite and discrete state and action spaces, S and A . The actualization values of the state-action value function are shown in Eq. (4):

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha [R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)] \quad (4)$$

where Q is the state-action value function, which can be stored in a tabular way, $s_t \in S$ is the system state at time t , $a_t \in A$ is the action taken at time t , α is the learning rate, and

γ is the discount factor. Regardless of which policy is used, the learned action-value function Q directly approximates the optimal action-value function Q^* . Once Q^* is known through interactions, then the optimal policy $\pi^*(s)$ can be obtained directly through:

$$\pi^*(s) = \max_a Q(S, a) \quad (5)$$

It is proven that *Q-learning* converges to optimal action values with probability one as long as all actions are sampled repeatedly in all states and action values are represented discretely (Watkins, 1989). As we will develop in the following section, in our proposal, we embed an RL formulation for optimal setting up the parameter r_1 of the optimization algorithm.

Metaheuristic DSCA using RL-optimized parameterization laws for MH optimization

As we mentioned earlier, our proposed approach focuses on the utilization of simulation, optimization, and RL techniques to address MH challenges within a manufacturing facility. Following, the performance of the proposed approach will be tested in a case study, encompassing scenarios ranging from less to more complex. This evaluation aims to assess the proposal's effectiveness across various realistic situations, providing insights into its practical applicability and adaptability. Finally, we will test the algorithm's effectiveness in reducing waste in the system.

Reinforced discrete sine cosine algorithm

MH optimization in production systems refers to efficiently moving raw materials, work-in-progress (WIP), and finished goods within a production facility during manufacturing. The MH problem commonly requires decisions that involve choosing a specific action from a finite set of options or choices, where the variables involved can only take on discrete values.

As we stated in section "Optimization", the DSCA could be used to solve the MH problem. Although SCA has been utilized in several previous works, DSCA is a novel method that has received little attention and has yet to be used to address MH problems. The pseudocode shown above in Algorithm 2 outlines the main steps of the DSCA. However, as we will see in section "Results and discussions", the standard DSCA underperforms our proposed algorithm to solve the MH problem. This is mainly due to its convergence being affected by the established values of the parameters r_i in Eq. (3).

In order to improve the DSCA performance, we have been inspired by another algorithm, based on SCA, called

QLESCA, recently proposed by Hamad et al. (2022). QLESCA includes a *Q-learning* formulation for the SCA parameters selection. The algorithm *Q-learning* helps each QLESCA agent to move adaptively in the search space according to its performance, where the reward (or penalty) is given based on the agent’s achievement. Each agent has a *Q* – table, where the parameters r_1 and r_3 are obtained based on the chosen state and action. Algorithm 3 outlines the QLESCA original pseudocode. Although QLESCA improves the performance of SCA, it was developed to deal with optimization problems in continuous domains, and the MH optimization problem requires discrete solutions. So, taking Algorithms 2 and 3 into account, we develop the proposed Reinforced Discrete Sine Cosine Algorithm, referred as *R*DSCA* outlined in the pseudocode shown in Algorithm 4.

Algorithm 3 Q-learning embedded Sine Cosine Algorithm

Set T = maximum number of iterations
Initialize a set of search agents (X) and the corresponding Q-tables
Evaluate each agent by the objective function and save the solutions
Update the best solution obtained so far ($P = X^*$)
While $t < T$:
 For each agent:
 Calculate the current state S_t of each agent
 Choose an action A_t and act
 Update r_1, r_2, r_3 and r_4
 Update search agents positions using Eq. (1)
 Update the best solution of each agent:
 If the solution is better: Reward = 1
 Else Reward = -1
 Update Q-tables using Eq. (4)
Return the best solution obtained so far as the global solution

An important novelty of our proposed *R*DSCA* is that we have developed a mechanism for finding an optimal way to learn to increase the r_1 parameter. The coefficient r_1 is known as a transition control parameter, and its main function is to strike a balance between diversity and convergence during the search procedure (Gupta et al., 2022). The value of this parameter should preferably be increased across iterations to provide a transition from exploration to exploitation of the search space. As we see below, including a learning mechanism for r_1 selection substantially improves the algorithm’s performance. In this way, the transition control parameter r_1 could be updated following some behavior law. However, the possibility of learning a behavior law introduces an additional degree of freedom to the algorithm that can be taken advantage of to improve its performance. That is, instead of fixing a function that describes that behavior in advance, we can learn to build a monotonic increasing function from a finite set \mathcal{R} of more elementary functions $r_{1,j} \in \mathcal{R}, j = 1, \dots, N$ using the *Q-learning* algorithm. So, in order to implement a tabular version of the *Q-learning* algorithm in a simple

Algorithm 4 *R*DSCA*

Set T = maximum number of iterations
Initialize a set \mathcal{X} of search agents and the corresponding Q_i tables for each agent $X_i \in \mathcal{X}$
Evaluate $f(X_i) \forall X_i \in \mathcal{X}$ and save the evaluations results, f_i , such that $f_i = f(X_i) \in \mathcal{F}, i = 1, 2, \dots, |\mathcal{X}|$
Update the best solution obtained so far, such that $X_d = \max_{X \in \mathcal{X}} \{f_i \in \mathcal{F}\}$
While $t < T$:
 For each $X_i^t \in \mathcal{X}$:
 If $p < \varepsilon$:
 Randomly set search agent position X_i^t and evaluate $f(X_i^t)$
 Else:
 Obtain the current state s_t as the nearest $s^* \in S$ to δ_i^t computed by Eq. (6)
 With π_i^* obtain $r_1 = \pi_i^*(s_t) = \max_r Q(S, r)$ (Eq. (5))
 Update r_2 and r_3 randomly
 Compute C_i^t as in Eq. (3)
 Update search agent position X_i^t using Eq. (2)
 Evaluate $f(X_i^t)$
 Calculate the reward comparing $f(X_i^t)$ whit the best agent solution
 Update Q_i using Eq. (4)
 Return the best solution obtained so far as the global solution X_d

way, we must have a set of finite states. Still, to convert the r_1 selection problem into a purely discrete finite MDP, defining a strategy to deal with discrete states remains. Therefore, we must define the states of the MDP. We define the space of states S in the domain $[0, 1]$, such that each s_τ is an interval contained in S , such that $s_\tau \subset S$ with $s_\tau = 1, \dots, m$ and the states s_1, \dots, s_m , are congruent intervals into the domain of S . Additionally, we define the current relative distance δ_i^t between the solutions found and the optimal solution as:

$$\delta_i^t = \frac{X_d^t \ominus X_i^t}{l} \tag{6}$$

where the numerator indicates the number of feature differences between the solution found and the optimal solution, and the denominator indicates the length of these solution vectors. So, the current state s_t will be that $s^* \in S$ closest to the current value δ_i^t . That is, for each agent at each instant, δ_i^t will be calculated, and the corresponding state will be found. In this way, using this strategy, we obtain a finite-state machine to describe the MDP, and consequently, the *Q-learning* algorithm could be used to obtain the optimal function for r_1 according to the current state s_t , following the state-action value function given in Eq. (4).

Returning to the step-by-step explanation of Algorithm 4, we see that it starts from a random solution. Then, each agent is evaluated by means of the objective function, and the proposed solution is saved. Then, the best solution is obtained. Continuing with the central part, which is the population evolution, as we can see, sometimes, with a probability ε , a random solution is taken (line 7, 8) and is evaluated through

simulation, following on the other hand with the evolution of the algorithm.

It can be seen in lines 10–17 the Q -learning section, where for each agent is obtained the current state and the action execution, as explained above. Once the value of r_1 is calculated, r_2 and r_3 are randomly updated. Using Eq. (2) and (3), the agent's positions are updated, and their function's evaluation results are saved. The exact elements are kept to update the agent's position, and the order of the different features is updated randomly to generate a new solution. For the Q -learning reward, we compare the best solutions obtained for each agent with the current agent solution, and then the Q -tables are updated. Finally, the best solution for the population is returned. For a better comprehension of the developed algorithm, Fig. 1 shows a flowchart of it.

Case study

Problem definition

In the present work, the problem presented in Gola and Kłosowski (2019) is adapted as a case study. In particular, the benchmark problem has a high complexity where an MH problem is stated for a manufacturing plant, and it is solved by optimizing the routes of automated guided vehicles (AGV). The presented problem has a manufacturing system comprising 20 workstations and an autonomous vehicle.

The transport system of the case study has two fundamental tasks at certain times: delivery of parts to the workstations and collection of parts from the workstations. There is only one point of departure and arrival for the vehicle: the dispatch station. Once the vehicle has departed from this point, the route remains fixed, and there are no further changes.

Several aspects of the original problem are maintained in this work. For example, the plant layout and the stations' working times. Also, the load sizes for each station. And the initial stocks are equal to the load at each station. These data can be consulted in Table 1. The dispatch station is located at point (0, 0).

The final objective of the study is to minimize the distance traveled by the vehicle on each route that must be developed. Before this, it is decided by predetermined fixed rules to which stations the vehicle should travel. It is also determined whether the transport should carry or remove materials from these stations. This will depend on the state of the stations, that is to say, the time remaining until the stock runs out and the amount of final stock in each.

R^*DSCA Setup

For this particular problem presented in the previous subsection it will be defined here the main characteristics of our proposal. As mentioned, Q -learning finds the best val-

Table 1 Case study data (Gola and Kłosowski, 2019)

Workstation	Initial Stock (u)	Time machining a load (s)
1	50	4000
2	60	4200
3	20	4000
4	20	4800
5	10	4000
6	40	4000
7	40	4000
8	30	3900
9	20	4000
10	10	4000
11	20	4000
12	50	5000
13	30	3750
14	10	4000
15	20	4000
16	30	6600
17	10	4000
18	50	4000
19	25	3500
20	25	3500

Table 2 States for R^*DSCA setup

S_i	1	2	3	4	5
S_i Values	[0; 0.2]	(0.2; 0.4]	(0.4; 0.6]	(0.6; 0.8]	(0.8; 1]

ues of $r_{1,j}$ for each iteration. Four increasing functions were chosen for the parameter $r_{1,j}$. Three of them were worked in the interval (0, 1) and the rest in the interval (0, 0.5). The formulas are shown below, and Fig. 2 shows the graphs corresponding to an interval of 30 iterations. Therefore, the actions A_i of the algorithm will be the different functions $r_{1,j}$ whit $j = 1, \dots, 4$.

$$r_{11} = \frac{t}{T} \quad (7)$$

$$r_{12} = \frac{t}{2T} \quad (8)$$

$$r_{13} = 1 - \epsilon^{-(t^{0.5})} \quad (9)$$

$$r_{14} = \frac{t^2}{T^2} \quad (10)$$

Regarding the necessary space of states S in the domain [0, 1] to apply the RL, five states were defined based on the current relative distance δ_i^t (Eq. (6)), such that $s_\tau \subset S$ with $s_\tau = 1, \dots, 5$. The states can be seen in Table 2. Therefore, in our proposal, each Q -table has five states and four actions.

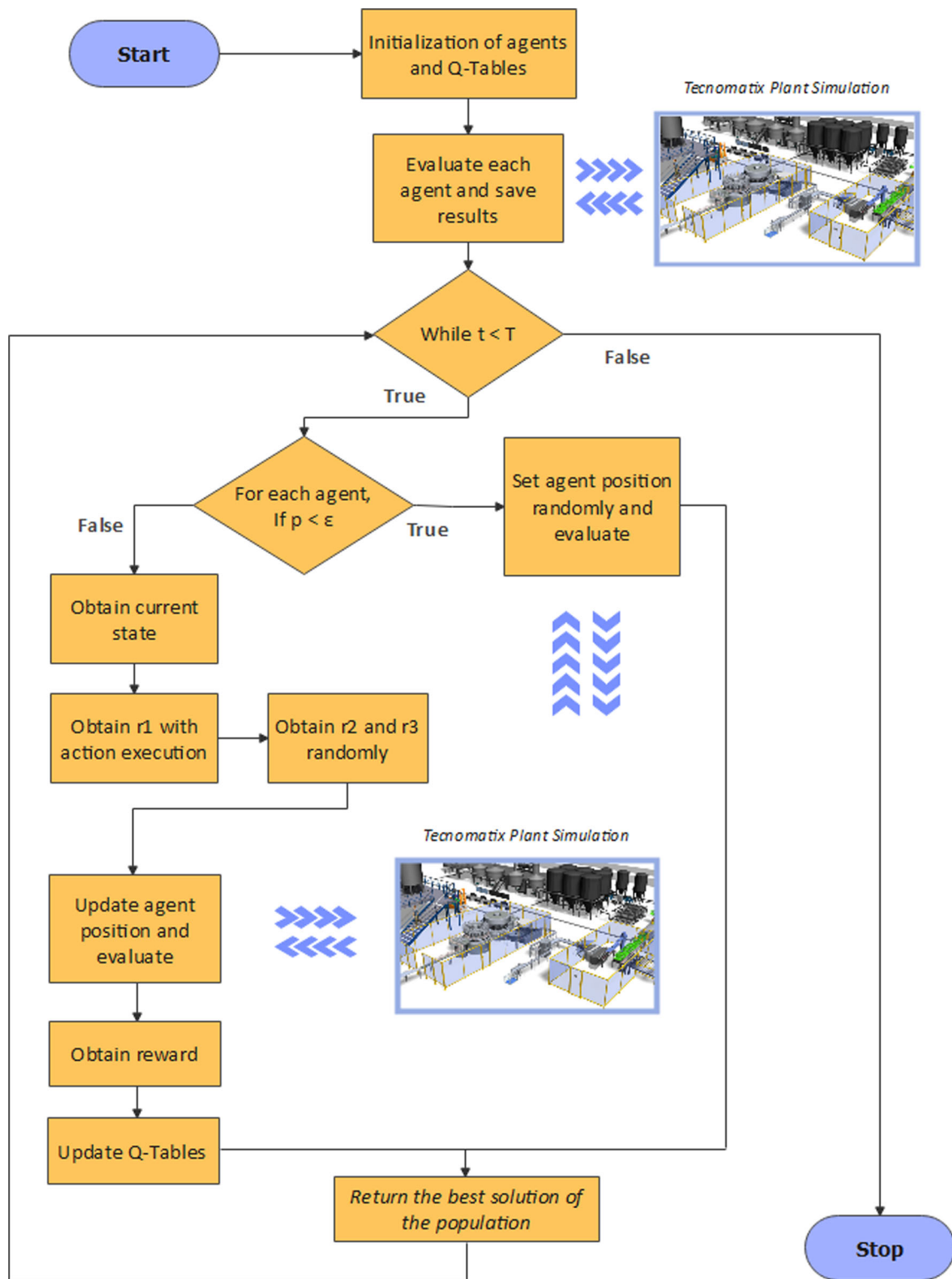


Fig. 1 R*DSCA flowchart

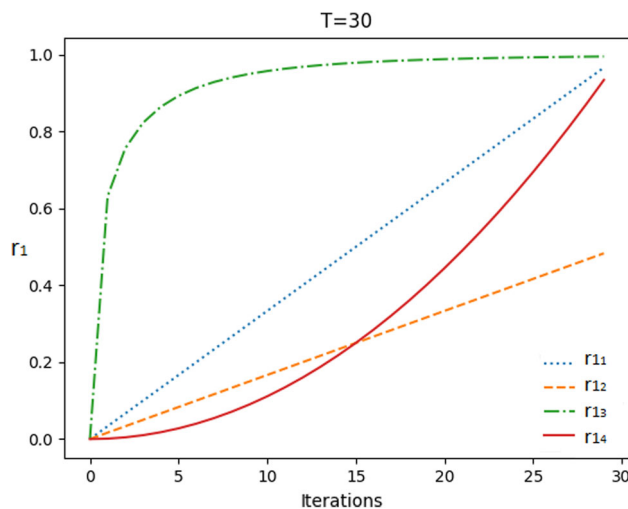


Fig. 2 Functions to apply for the parameter r_1

Regarding the reward, what is detailed in Algorithm 3 was maintained; that is, if the solution found by the agent is better than the one obtained so far, the reward is 1; otherwise, it is -1.

Finally, ε was set at 0.20. And, when applying Q -learning, the internal parameters α and γ were set to values of 0.1 and 0.9, respectively. These values are the most commonly used empirically in the literature. In addition, the agent was trained following an ϵ -greedy exploration strategy, with a decay of ϵ with the course of the episodes from $\epsilon = 1$ to a minimum value $\epsilon = 0.01$, to improve the exploration-exploitation strategy.

Computational environment setup

For the modeling of the system and environment for the execution of the algorithms, it was decided to use the software *Tecnomatix Plant Simulation* (TPS) from Siemens. This software allows us to easily and quickly model and visualize productive systems. It is an object-oriented software that can customize and adapt these objects by adding specific programming. Figure 3 shows the model built in TPS for the problem in question. As can be seen, the routes are fixed, and the stations have a single place for loading and another for unloading.

The proposed optimization algorithm R^*DSCA was developed in Python language. In addition, the computational implementation uses the software architecture *SimulAI* (Perez Colo et al., 2020) previously developed in the group (Saavedra Sueldo et al., 2022, 2021). This architecture was developed by our team earlier and allowed us to interconnect different entities and exchange information between them. *SimulAI* is a library that facilitates the use of AI techniques to optimize variables derived from the Simulation of Flexible Manufacturing Systems. The objective is to improve the

decision-making process in organizations, making it more efficient and safe. In this case, we mainly use *SimulAI* to encapsulate the TPS simulations with our computational algorithmic formulation.

The Python code comprises a general simulation of the plant and similar simulations that are opened and closed by running a specific path in the optimization algorithm to test a point solution. The overall simulation for the first phase of the problem (deciding where to go) is continuously monitored, and the algorithms are used to find the best possible route sequence.

The complexity of the developed model is $O(T * |X|)$, with T the number of iterations and $|X|$ the number of initial solutions. The algorithmic developments of this work, as well as the details of the case study, are available in the digital repository <https://github.com/carosaav/SimulAI/tree/master/RDSCA%20algorithm>

Results and discussions

Initially, in section “ R^*DSCA assessment” to show the performance of our proposed R^*DSCA optimization algorithm, we consider four sub-problems (or scenarios) with different complexity levels based on the problem presented in section “Problem definition”. We start addressing the least complex sub-problems, then another with intermediate complexity, and finally, a more complex sub-problem than the other two are addressed. When dealing with tractable problems with a few workstations, we can find the global optimal solutions by simulating all possible solutions. In this way, we take advantage of having the plant simulator, which allows us to simulate all possible solutions and accurately identify the optimal one. Thus, the solution that gives the best objective function value will be the optimal one. Then, this optimal solution will be used to compare the solution found using our proposal.

Once we demonstrate the performance of our proposal, in section “ R^*DSCA assessment for waste reduction”, we address the whole problem presented in section “Problem definition” and illustrate the applicability of our proposed algorithm for a large case considering waste reduction. However, finding the optimal global solution using intensive computational simulations is impracticable when faced with complex scenarios with several workstations since extremely large (or prohibitively) computational processing times will be required. As we have previously stated, the main problem has 20 workstations. Therefore, this gives rise to $2.432902e+18$ ($20!$) possible routes (feasible solutions).

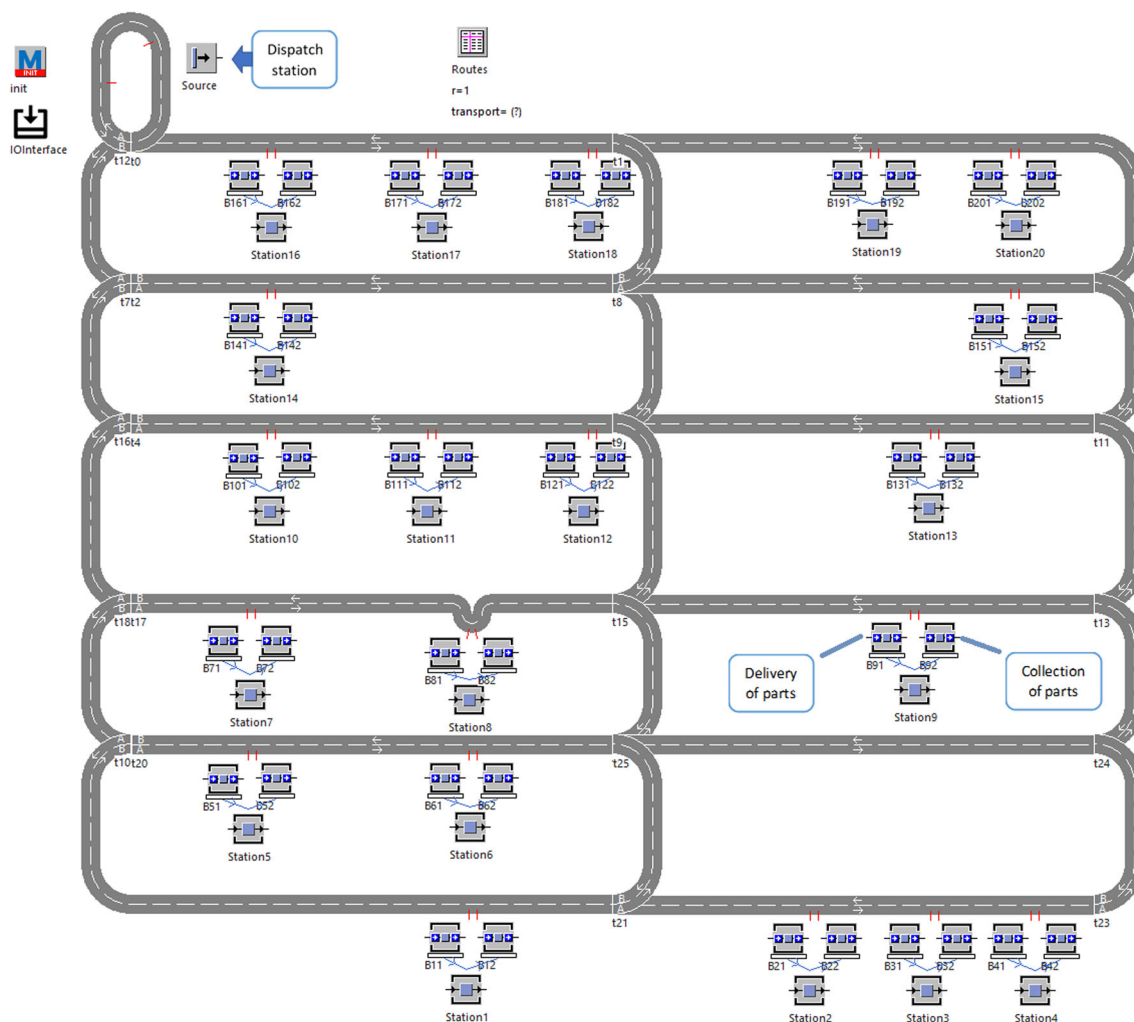


Fig. 3 Simulation model of the case study

R*DSCA assessment

As detailed above, we consider four sub-problems explained below to assess the performance of *R*DSCA*. Following, in section. “Results for *R*DSCA*”, we show the results obtained when *R*DSCA* is used to solve these cases. Then, in section “Performance comparison”, we compare performance against another comparable methodology, the *DSCA*, for the same cases.

Results for *R*DSCA*

Sub-problem A

To begin with, the initial sub-problem is simple, made up of only five stations out of the 20 that make up the complete original system. In this case, only workstations #1, #4, #6, #15, and #16 are considered, and Fig. 4 shows the configuration of the resulting system. To analyze the performance of the proposed algorithm, we designed four testing experi-

ments for this sub-problem. The results of each experiment are summarized in Table 3.

In order to determine how the initialization affects the solution found by the algorithm, we have run experiments with a fixed initial population and with a random one (mentioned in the 2nd column of Table 3). In both cases, the initial population includes six initial feasible solutions. In addition, to determine how the number of iterations affects the solution found by the algorithm, we have run experiments with different numbers of iterations, *T*, given in the 3rd column of Table 3. As previously explained, for this problem, the optimal theoretical solution, denoted as *O_{TS}*, is found by simulation of all feasible solutions. It is given in the 4th column of Table 3.

For each experiment, due to the stochastic nature of the proposed *R*DSCA*, the value \bar{D} , shown in the 5th column of Table 3, corresponds to the average of the results (*D_i*) obtained for *m* = 10 repetitions of such an experiment. It is computed as in Eq. (11). The optimal solution, *D**, found by

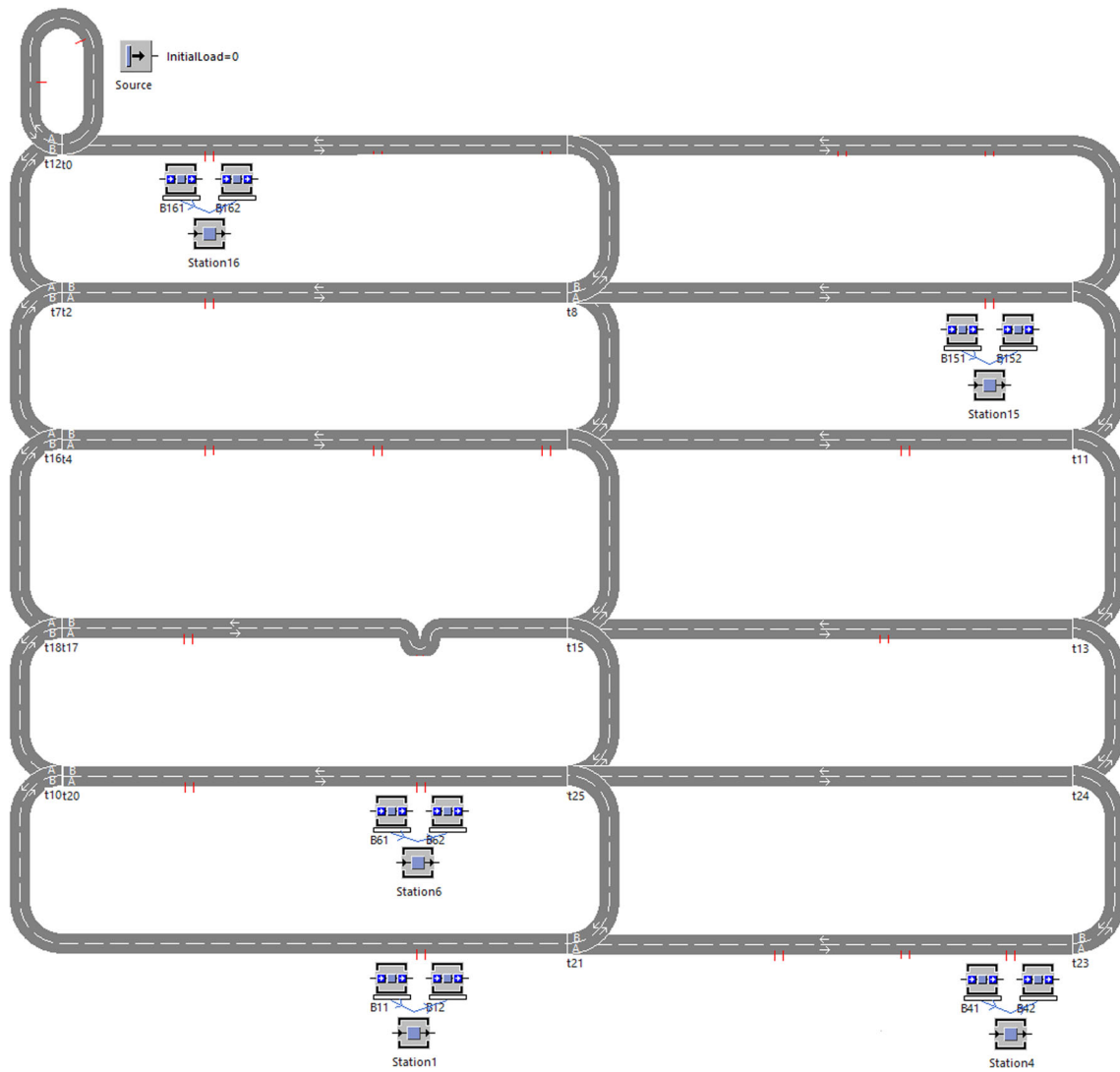


Fig. 4 Simulation model of sub-problem A

Table 3 R* DSCA experiments on sub-problem A

Experiment	Initialization	T	O_{ts}	\bar{D}	D^*
1	fixed	30	374.3 m	386.6 m	374.3 m
2	fixed	50	374.3 m	380.6 m	374.3 m
3	random	30	374.3 m	385.0 m	374.3 m
4	random	50	374.3 m	375.9 m	374.3 m

the algorithm is given in the last column of Table 3, and it is obtained as in Eq. (12).

$$\bar{D}_t = \frac{\sum_{i=1}^m D_{t_i}}{m}; \quad t = 1 \dots T \tag{11}$$

$$D_t^* = \max_j \{D_{t_j}\}; \quad j = 1 \dots m \tag{12}$$

It can be verified that, for this simplified sub-problem, the algorithm always finds the optimal solution and that most of the tests are close to this solution since the average solution of each one is very close to the optimal one.

Sub-problem B

In this case, we address a variant with the same complexity as in the previous case, i.e., we take into account only five workstations too, but now we take into account the workstations #2, #7, #9, #14, and #19. Therefore, the resulting shop floor setup is as in Fig. 5. In the same way, as in the previous case, we repeat the same experiments, and the obtained results are given in Table 4. Once again, the successful performance of the algorithm can be seen in all the tests since Algorithm 4 consistently achieves the optimal solution, that is, $D^* = O_{ts}$. Also, another issue that can be observed is that the initialization of the algorithm does not affect its perfor-

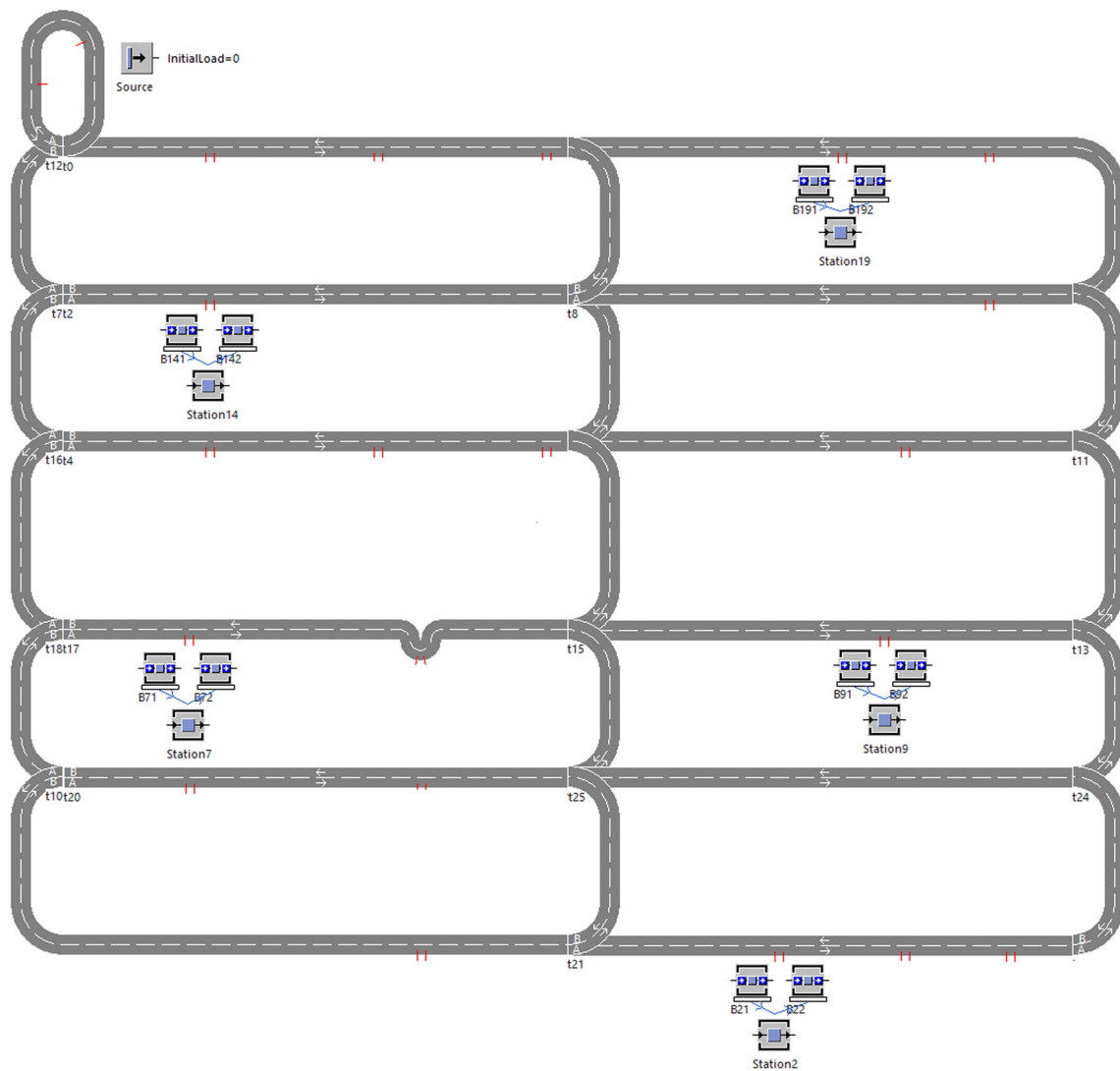


Fig. 5 Simulation model of sub-problem B

Table 4 R* DSCA experiments on sub-problem B

Experiment	Initialization	T	O_{rs}	\bar{D}	D^*
1	fixed	30	382.1 m	386.7 m	382.1 m
2	fixed	50	382.1 m	382.2 m	382.1 m
3	random	30	382.1 m	384.0 m	382.1 m
4	random	50	382.1 m	382.2 m	382.1 m

mance. Therefore, we will consider only random initiations for the following experiments to avoid bias.

This is to highlight that in the case of a flexible production system, the periodic reconfiguration of production facilities is a frequent issue since the facilities must be reconfigured to meet different production requirements.

For both sub-problem A and sub-problem B, the proposed optimization algorithm successfully reached the optimal

solution of the problem. This demonstrates that, in addition to arriving at the optimal solution for MH, our proposal is appropriate for use in these types of reconfigurable production systems. Next, to demonstrate the successful performance of our proposal, we will present other considerably more complex cases.

Sub-problem C

This sub-problem is more complex than the previous ones, and it is composed of seven workstations, which are #2, #5, #7, #9, #11, #14, and #19, and the resulting shop floor setup is shown in Fig. 6.

This sub-problem is larger than the previous ones because the space of feasible solutions increases. Consequently, giving the algorithm a greater possibility of exploration is logical. Therefore, we include three additional experiments by increasing the maximum number of iterations, T.

Table 5 R^*DSCA experiments on sub-problem C

Experiment	T	O_{ts}	\bar{D}	D^*
1	30	504.1 m	534.2 m	504.1 m
2	50	504.1 m	525.5 m	504.1 m
3	100	504.1 m	517.2 m	504.1 m
4	200	504.1 m	515.0 m	504.1 m
5	500	504.1 m	509.4 m	504.1 m

Table 6 R^*DSCA experiments on sub-problem D

Experiment	T	\bar{D}	D^*
1	50	724.1 m	677.2 m
2	100	684.0 m	653.1 m
3	150	671.2 m	611.3 m
4	200	658.0 m	598.9 m
5	500	613.5 m	566.4 m

Table 5 summarizes the results obtained. As can be seen, the algorithm always finds the optimal solution, $D^* = O_{ts}$, even when it runs with low iterations. As in the previous cases, in the 4th column of Table 5, it can be seen that as T increases, the algorithm's performance improves since the mean of the found optimal solutions decreases.

Sub-problem D

To finish with the performance analysis of the algorithm, we are going to consider a complex problem in which we have ten workstations: #1, #2, #4, #8, #9, #11, #12, #14, #15, and #19 and the production system setup results as in Fig. 7. Unlike the previous cases, we do not have the theoretical optimal solution O_{ts} . Thus, we analyze the algorithm's performance as the number of iterations T increases.

Table 6 shows the obtained experimental results. It is observed that the greater the number of iterations T , the smaller the average distances obtained. Therefore, the optimal solutions found D^* are also better.

As we have hinted, the proposed simulation-based meta-heuristic algorithm, R^*DSCA , contributes to determining MH policies for flexible production systems, where the reconfiguration of the facilities can be carried out frequently to meet different production requirements.

Performance comparison

Following, we continue with a performance comparison of our proposal against another comparable methodology to address sub-problems A to D. Specifically, we use the original DSCA. For the DSCA application (Gupta et al., 2022), the difference in features of the vectors implies the discrepancy in the sequence of the routes tested. In addition, with a cer-

Table 8 Experiment deviations

Experiment	DSCA deviation	R^*DSCA deviation
A1	2.175%	3.295%
A2	2.766%	1.695%
A3	3.730%	2.860%
A4	3.295%	0.435%
B1	2.156%	1.182%
B2	1.176%	0.001%
B3	3.525%	0.491%
B4	1.664%	0.004%
C1	8.178%	5.962%
C2	5.737%	4.243%
C3	7.018%	2.600%
C4	2.862%	2.161%
C5	0.785%	1.047%

tain probability ($p = 0.20$), the choice of random solutions is added to increase the exploration of feasible solutions.

In the same way, as in the previous case, all the tests were repeated ten times with six initial solutions for each experiment. Summarizing, below in Table 7 are the results of the experiments with the DSCA for these four different sub-problems. It is verified that in almost all cases, DSCA finds the optimal solution, but the R^*DSCA achieves better average results, demonstrating better performance than DSCA.

Finally, to carry out this performance comparison, we have only considered those experiments for which we have the optimal solution O_{ts} . So, for each experiment carried out for each sub-problem, in Table 8, we computed the deviation between the obtained \bar{D} by DSCA and R^*DSCA , respectively, and the optimal solution O_{ts} .

Analyzing the results presented in Table 7 and Table 8, it can be inferred that the performance of our proposed R^*DSCA is higher than DSCA, reaching fewer deviations in 11 of the 13 first cases with known optimal solutions.

In the case of sub-problem D, for the comparison, we build some graphs that show the convergence curves for both algorithms (Fig. 8) used to find an optimal solution for this stated problem. By having a random start to the algorithm, there are different starter points, but a convergence at smaller distances can be observed in R^*DSCA in all the figures.

R^*DSCA assessment for waste reduction

Summarizing up to this point, we have tested the algorithms on different sub-problems with different configurations and verified their effectiveness. We have demonstrated that the DSCA with RL parameterizing laws has the best performance. Now, another testing stage begins, where experiments

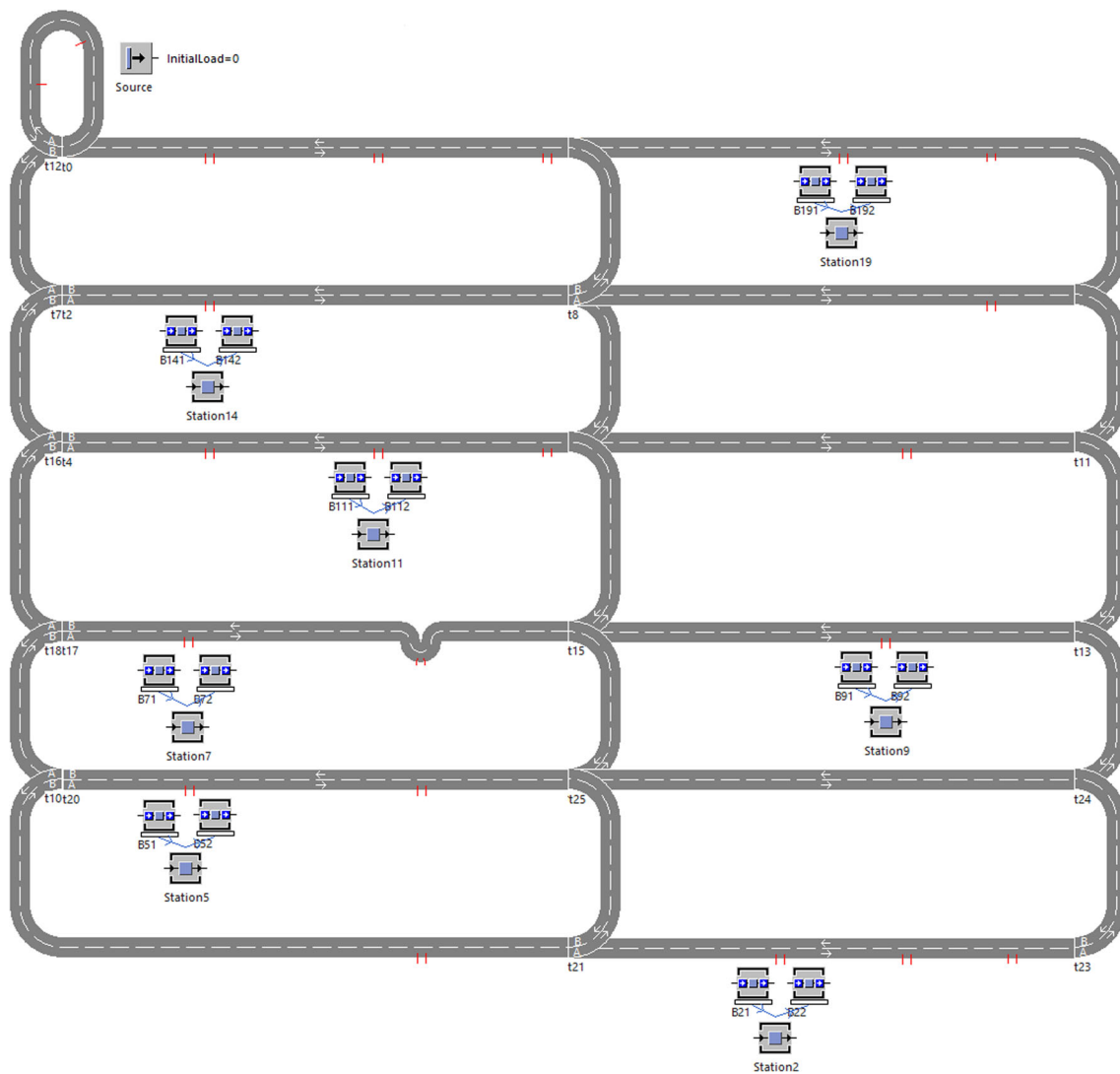


Fig. 6 Simulation model of sub-problem C

are run for the complete system of 20 stations (Fig. 3). Therefore, this problem is even more complex than the previous ones and has too many feasible solutions. In addition, we verify that the developed proposal not only optimizes the distances but also helps to reduce other Lean wastes.

For this stage, several rules based on different transport route sequences that organizations may use in their daily operations are tested against the performance of R^*DSCA . The objective will be to verify if the R^*DSCA optimization of MH impacts the Lean waste of the system, specifically transportation, waiting, and inventories, which we will check in the simulator.

The first three cases have no built-in optimization system, so the transport is directed in a given order to fulfill the required route following a rule. In **case 1**, the order of the sequence or scheduling rule is random. This would be the case for an organization that does not provide any plan-

ning for the transport of materials due to a lack of knowledge or resources. In **case 2**, the vehicle is directed in ascending numerical order to the stations, following the layout. In **case 3**, transport maintains the priorities given by the initial part of the system. This part is mentioned and explained in section “Case study”, the scenario with more logic for handling materials because it considers the state of the stations and their buffers. Finally, **case 4** is the application of our proposal R^*DSCA with $T=100$. The limit for the number of workstations to be covered by the transport on the same trip remains ten.

Ten repetitions of cases 1 and 4, which contain randomness, are carried out. Cases 2 and 3 are fixed for a given initial situation. It is verified in the TPS software that the distances traveled by the vehicle, the time that the stations must wait for parts, and the remaining stock at the end of the simulations are lower with the developed algorithm. Therefore,

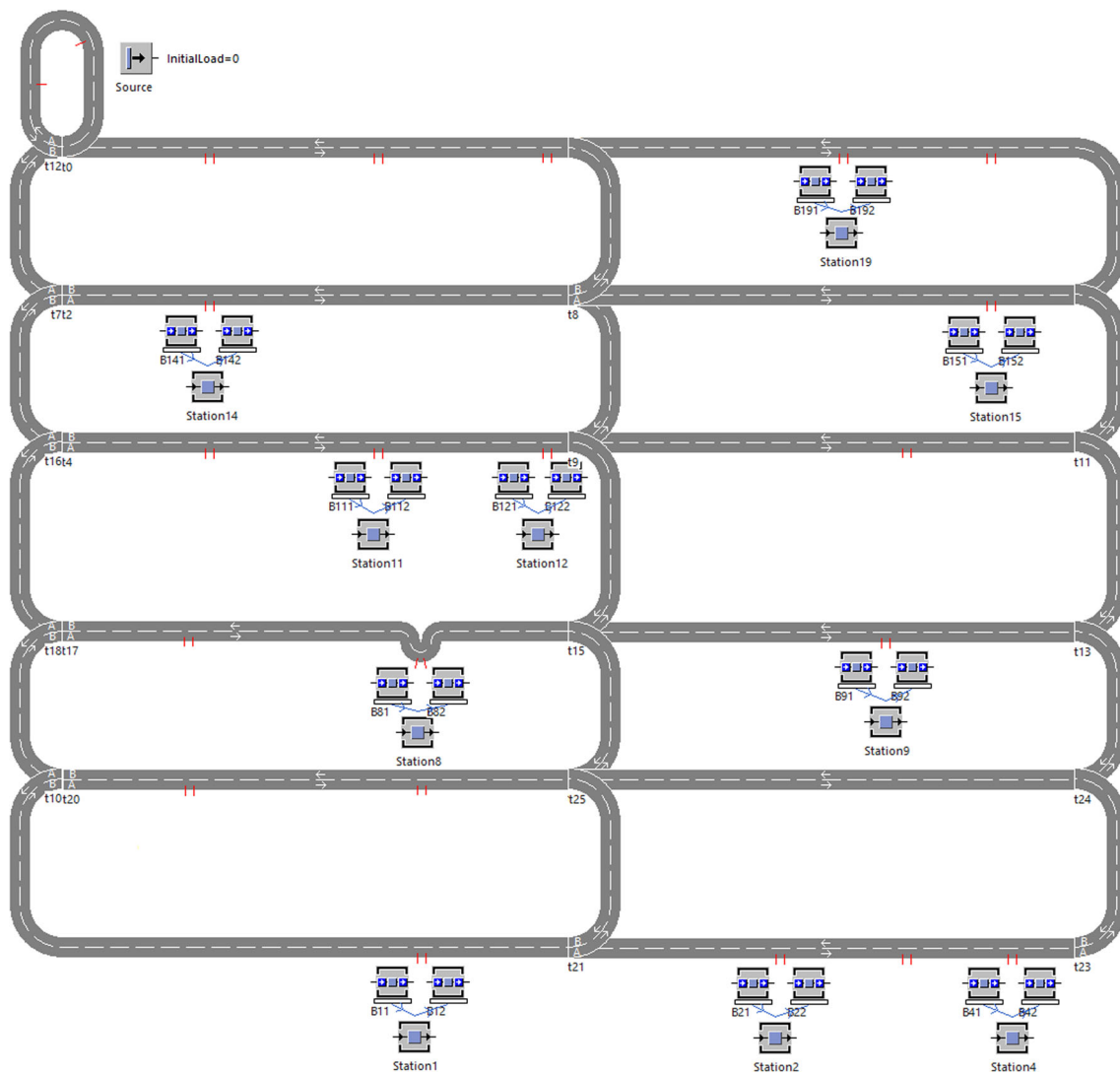


Fig. 7 Simulation model of sub-problem D

it is possible to reduce three significant wastes in any production system, significantly impacting costs and business productivity.

Tables 9 and 10 show the best results obtained for each case. The production period tested in the first table is 2 h for each test, while in the second table, it is 8 h. In all cases, we start from the same initial situation to be able to compare. The last column of these tables shows the waiting time of the system station that most had to wait from the twenty stations.

Tables 11 and 12 show the Lean waste reduction percentages when comparing the first three cases against our proposal R^*DSCA . In a few tests and with a small number of algorithm iterations ($T = 100$), minimizing such waste is significant in both cases.

Finally, to conclude this section and with the experiments carried out, the general 8-hour production period simulation was tested with the four rules but no longer with restrictions

Table 9 Results for 2-hour production period with $T = 100$

	Total distance traveled	Total final stock	Maximum waiting time at station
Rule 1	3709.01m	681 u	11.85%
Rule 2	3453.09m	703 u	11.93%
Rule 3	3670.53m	681 u	11.36%
R^*DSCA	2923.67m	660 u	9.46%

on the number of stations that can cover the transport in a single tour. In this scenario, the application of R^*DSCA is performed with $T=200$. The results are shown in Tables 13 and 14. Once again, the algorithm's high performance and waste reduction results are confirmed, mainly in the waiting time of stations.

Table 7 DSCA experiments on the four sub-problems

Sub-problem	Experiment	Initialization	No. of stations	T	O_{ts}	\bar{D}	D^*
A	1	fixed	5	30	374.3 m	382.4 m	374.3 m
A	2	fixed	5	50	374.3 m	384.6 m	374.3 m
A	3	random	5	30	374.3 m	388.3 m	374.3 m
A	4	random	5	50	374.3 m	386.6 m	374.3 m
B	1	fixed	5	30	382.1 m	390.4 m	382.1 m
B	2	fixed	5	50	382.1 m	386.6 m	382.1 m
B	3	random	5	30	382.1 m	395.6 m	382.1 m
B	4	random	5	50	382.1 m	388.5 m	382.1 m
C	1	random	7	30	504.1 m	545.4 m	517.3 m
C	2	random	7	50	504.1 m	533.1 m	504.1 m
C	3	random	7	100	504.1 m	539.5 m	504.1 m
C	4	random	7	200	504.1 m	518.6 m	504.1 m
C	5	random	7	500	504.1 m	508.1 m	504.1 m
D	1	random	10	50	–	744.7 m	678.4 m
D	2	random	10	100	–	708.6 m	659.2 m
D	3	random	10	150	–	691.3 m	659.7 m
D	4	random	10	200	–	669.3 m	631.4 m
D	5	random	10	500	–	623.3 m	571.5 m

Table 10 Results for 8-hour production period with T = 100

	Total distance traveled	Total stock	final Maximum time at station	waiting
Rule 1	18422.83 m	1064 u	16.39%	
Rule 2	15907.94 m	831 u	15.45%	
Rule 3	18160.87 m	1072 u	10.45%	
R^*DSCA	15902.96 m	725 u	8.50%	

Table 13 Results for 8-hour production period with T = 200

	Total distance traveled	Total stock	final Maximum time at station	waiting
Rule 1	18728.23 m	808 u	29.33%	
Rule 2	16248.55 m	726 u	17.88%	
Rule 3	19397.50 m	1014 u	29.05%	
R^*DSCA	15782.52 m	705 u	11.74%	

Table 11 Waste reduction for 2-hour production period with T = 100

	Transportation	Stock	Waiting
Rule 1 vs. R^*DSCA	21.17%	3.08%	20.17%
Rule 2 vs. R^*DSCA	15.33%	6.12%	20.67%
Rule 3 vs. R^*DSCA	20.35%	3.08%	16.70%

Table 14 Waste reduction for 8-hour production period with T = 200

	Transportation	Stock	Waiting
Rule 1 vs. R^*DSCA	15.73%	12.75%	59.97%
Rule 2 vs. R^*DSCA	2.87%	2.89%	34.34%
Rule 3 vs. R^*DSCA	18.64%	30.47%	59.59%

Table 12 Waste reduction for 8-hour production period with T = 100

	Transportation	Stock	Waiting
Rule 1 vs. R^*DSCA	13.68%	31.86%	48.15%
Rule 2 vs. R^*DSCA	0.03%	12.76%	45.00%
Rule 3 vs. R^*DSCA	12.43%	32.37%	18.63%

Advantages of the developed R^*DSCA

It is worth highlighting some central aspects of our proposal. One of the main contributions of this work is the development of an optimization algorithm to deal with optimization

problems involving discrete decision variables since many plant floor problems can be formulated as discrete decision-making problems.

In summary, our proposal stated a discrete formulation of the Sine Cosine Algorithm with an RL-embedded algorithm that allows for self-learning of the behavior law for the exploration/exploitation parameters. Including reinforcement learning to update a parameter provides more excellent performance to the algorithm while generating a greater degree of autonomy and independence from the intervention of a human expert. However, this could even be considered to

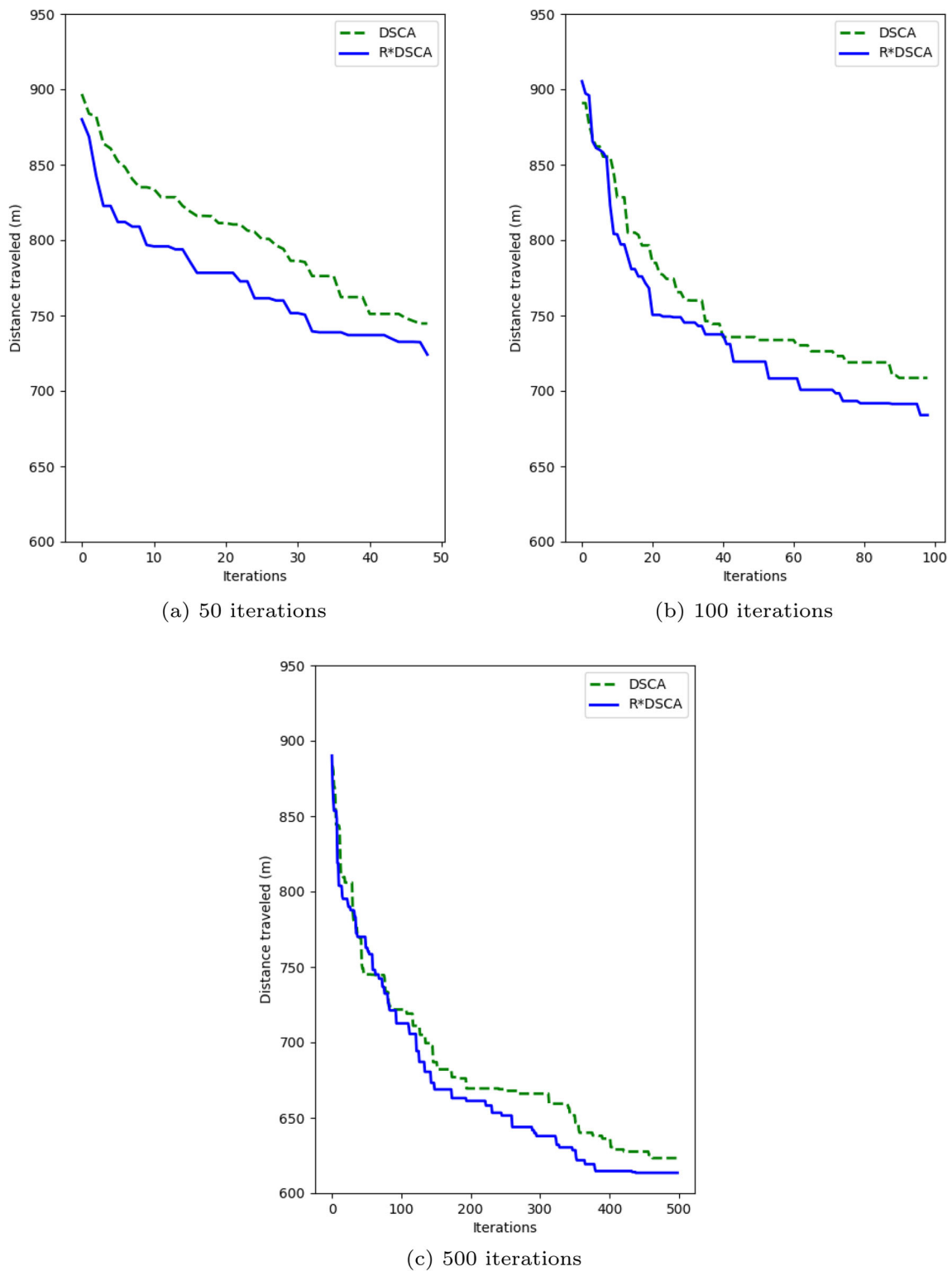


Fig. 8 Convergence curves \bar{D}_i comparison on sub-problem D

extend the idea to address the determination of other parameters not considered in this study.

On the other hand, a central contribution of our work is using a discrete event simulator like *Tecnomatix* since several previous works use graph theory for this type of problem,

providing less information. In addition, using a plant simulator would allow not only measurements of the roads but also other conditions, such as, to name a few, energy consumption, equipment failures, and others. In this way, involving the interconnection with a plant simulator paves the way to interact with a digital twin of the system in the future. In this sense, for example, we can more easily monitor system variables like waste in our case. In addition to having greater use in the industry and much development, these technologies allow migrating over time to a digital twin of the entire organization. This is a central issue for the future of Industry 4.0, including new industrial metaverses (Alimam et al., 2023).

Finally, there is a stretch relationship between lean manufacturing and material handling optimization algorithms. Lean principles provide the global philosophy for minimizing waste and continuous improvement. At the same time, optimization algorithms offer the tools and methodologies to implement and enhance these principles in the context of material handling and production processes. Together, they contribute to creating more efficient, flexible, and responsive manufacturing systems. The presented result demonstrated a high performance and waste reduction for the stated material handling problem.

Conclusion and future work

This work proposes a metaheuristic simulation-based optimization methodology to address the MH problem in dynamic environments. Our proposed approach integrates optimization, discrete event simulation, and artificial intelligence methods. A complex benchmark problem was addressed to assess the performance of our proposal deeply. In addition, based on that MH problem, more manageable sub-problems were stated to test and compare our proposal against other methodologies rigorously, and successful results were obtained.

Furthermore, using a plant simulator like *Tecnomatix* allowed us to study the complete production system and monitor Lean waste linked to MH. It was determined that our proposal contributes to reducing three wastes: transportation, inventory, and station waiting.

Another contribution of our work worth noting is our implementation since it uses a discrete-event plant simulator with the *SimulAI* architecture. This fact opens a course of action for future work since it allows, with a negligible effort, interchanging information between a decision-support system and a digital twin of a production system. This would enable us to deal with diverse environments, face even more dynamic problems, and address behavioral disturbances.

On the other hand, new trends in production sectors claim for developments supporting flexible production systems driven by customized product demand. The Lean 4.0

paradigm can significantly contribute to this requirement by promoting combining the concept of harnessing waste reduction with 4.0 technologies. Our proposal was conceived mainly to contribute to this issue since it was precisely raised to address the MH problem for flexible manufacturing systems. As previously emphasized throughout the manuscript, MH is one of the tasks that cause the most significant inefficiency in a production process. The time invested in moving materials within a shop floor is a task with no added value in the final product.

Also, as subsequent future work, it would be interesting to assess the performance of our proposal for other kinds of optimization problems related to the Lean 4.0 philosophy, like minimizing the environmental impacts of manufacturing activities in flexible plants. In addition, the developed proposal could be used in other engineering optimization problems, mainly those that can be formulated as optimal control problems with partial information on the environmental conditions.

Furthermore, another issue for future work is the configuration of the algorithm's hyperparameters, both those of SCA and those related to the Reinforcement Learning setup. Techniques such as Bayesian Optimization are possible new paths for this topic.

Data Availability Data will be made available on request.

Declarations

Conflict of interest The authors have no competing interests to declare that are relevant to the content of this article.

References

- Abdel-Baset, M., Zhou, Y., & Hezam, I. (2019). Use of a sine cosine algorithm combined with simpson method for numerical integration. *International Journal of Mathematics in Operational Research*, 14(3), 307–318.
- Adeodu, A., Maladzhi, R., Katumba, M. G. K. K., et al. (2023). Development of an improvement framework for warehouse processes using lean six sigma (dmaic) approach a case of third party logistics (3pl) services. *Heliyon*. <https://doi.org/10.1016/j.heliyon.2023.e14915>
- Akhand, M. A. H., Peya, Z. J., & Sultana, T., et al. (2015). Solving capacitated vehicle routing problem with route optimization using swarm intelligence. In *2015 2nd International Conference on Electrical Information and Communication Technologies (EICT)* (pp. 112–117). <https://doi.org/10.1109/EICT.2015.7391932>
- Akkari, A., & Valamede, L. (2020). Lean 4.0: A new holistic approach for the integration of lean manufacturing tools and digital technologies. *International Journal of Mathematical Engineering and Management Sciences*, 5, 851–868. <https://doi.org/10.33889/IJMEMS.2020.5.5.066>
- Alimam, H., Mazzuto, G., & Ortenzi, M., et al. (2023). Intelligent retrofitting paradigm for conventional machines towards the digital triplet hierarchy. *Sustainability*, 15(2). <https://www.mdpi.com/2071-1050/15/2/1441>

- Alkhorraif, A., Rashid, H., & McLaughlin, P. (2019). Lean implementation in small and medium enterprises: Literature review. *Operations Research Perspectives*, 6, 100089. <https://doi.org/10.1016/j.orp.2018.100089>
- Al-qaness, M. A. A., Elaziz, M. A., & Ewees, A. A. (2018). Oil consumption forecasting using optimized adaptive neuro-fuzzy inference system based on sine cosine algorithm. *IEEE Access*, 6, 68394–68402. <https://doi.org/10.1109/ACCESS.2018.2879965>
- Barata, J., & Kayser, I. (2023). Industry 5.0 - past, present, and near future. *Procedia Computer Science*, 219, 778–788. <https://doi.org/10.1016/j.procs.2023.01.351>. cENTERIS - International Conference on ENTERprise Information Systems / ProjMAN - International Conference on Project MANagement / HCist - International Conference on Health and Social Care Information Systems and Technologies 2022.
- Ben Moussa, F. Z., De Guio, R., Dubois, S., et al. (2019). Study of an innovative method based on complementarity between ariz, lean management and discrete event simulation for solving warehousing problems. *Computers & Industrial Engineering*, 132, 124–140. <https://doi.org/10.1016/j.cie.2019.04.024>
- Benitez, G. B., Ghezzi, A., & Frank, A. G. (2023). When technologies become industry 4.0 platforms: Defining the role of digital technologies through a boundary-spanning perspective. *International Journal of Production Economics*, 260, 108858. <https://doi.org/10.1016/j.ijpe.2023.108858>
- Benmessaoud, G. A., Yassine, M., Seyedali, M., et al. (2021). A comprehensive survey of sine cosine algorithm: variants and applications. *Artificial Intelligence Review*, 54, 5469–5540. <https://doi.org/10.1007/s10462-021-10026-y>
- Cifone, F. D., Hoberg, K., Holweg, M., et al. (2021). ‘Lean 4.0’: How can digital technologies support lean practices? *International Journal of Production Economics*, 241, 108258. <https://doi.org/10.1016/j.ijpe.2021.108258>
- Daoui, A., Karmouni, H., Sayyouri, M., et al. (2021). New robust method for image copyright protection using histogram features and sine cosine algorithm. *Expert Systems with Applications*, 177, 114978. <https://doi.org/10.1016/j.eswa.2021.114978>
- Dillinger, F., Bernhard, O., & Reinhart, G. (2022). Competence requirements in manufacturing companies in the context of lean 4.0. *Procedia CIRP*, 106, 58–63. <https://doi.org/10.1016/j.procir.2022.02.155>. 9th CIRP Conference on Assembly Technology and Systems.
- Dillinger, F., Kagerer, M., & Reinhart, G. (2021). Concept for the development of a lean 4.0 reference implementation strategy for manufacturing companies. *Procedia CIRP*, 104, 330–335. <https://doi.org/10.1016/j.procir.2021.11.056>. 54th CIRP CMS 2021 - Towards Digitalized Manufacturing 4.0.
- Duhamel, C., Lacomme, P., Quilliot, A., et al. (2011). A multi-start evolutionary local search for the two-dimensional loading capacitated vehicle routing problem. *Computers & Operations Research*, 38(3), 617–640. <https://doi.org/10.1016/j.cor.2010.08.017>
- Dündar, H., Soysal, M., Ömürgönülşen, M., et al. (2022). A green dynamic tsp with detailed road gradient dependent fuel consumption estimation. *Computers & Industrial Engineering*, 168, 108024. <https://doi.org/10.1016/j.cie.2022.108024>
- Furian, N., O’Sullivan, M., Walker, C., et al. (2015). A conceptual modeling framework for discrete event simulation using hierarchical control structures. *Simulation Modelling Practice and Theory*, 56, 82–96. <https://doi.org/10.1016/j.simpat.2015.04.004>
- Gambella, C., Ghaddar, B., & Naoum-Sawaya, J. (2021). Optimization problems for machine learning: A survey. *European Journal of Operational Research*, 290(3), 807–828. <https://doi.org/10.1016/j.ejor.2020.08.045>
- Ghodsian, N., Benfriha, K., Olabi, A., et al. (2022). Toward designing an integration architecture for a mobile manipulator in production systems: Industry 4.0. *Procedia CIRP*, 109, 443–448. <https://doi.org/10.1016/j.procir.2022.05.276>. 32nd CIRP Design Conference (CIRP Design 2022) - Design in a changing world.
- Gola, A., & Kłosowski, G. (2019). Development of computer-controlled material handling model by means of fuzzy logic and genetic algorithms. *Neurocomputing*, 338, 381–392. <https://doi.org/10.1016/j.neucom.2018.05.125>
- Golovianko, M., Terziyan, V., Branytskyi, V., et al. (2023). Industry 4.0 vs. industry 5.0: Co-existence, transition, or a hybrid. *Procedia Computer Science*, 217, 102–113. <https://doi.org/10.1016/j.procs.2022.12.206>. 4th International Conference on Industry 4.0 and Smart Manufacturing.
- Gouveia, L., Leitner, M., & Ruthmair, M. (2019). Layered graph approaches for combinatorial optimization problems. *Computers & Operations Research*, 102, 22–38. <https://doi.org/10.1016/j.cor.2018.09.007>
- Gupta, S., Zhang, Y., & Su, R. (2022). Urban traffic light scheduling for pedestrian-vehicle mixed-flow networks using discrete sine-cosine algorithm and its variants. *Applied Soft Computing*, 120, 108656. <https://doi.org/10.1016/j.asoc.2022.108656>
- Hamad, Q. S., Samma, H., Suandi, S. A., et al. (2022). Q-learning embedded sine cosine algorithm (qlasca). *Expert Systems with Applications*, 193, 116417. <https://doi.org/10.1016/j.eswa.2021.116417>
- Hu, H., Jia, X., He, Q., et al. (2020). Deep reinforcement learning based agvs real-time scheduling with mixed rule for flexible shop floor in industry 4.0. *Computers & Industrial Engineering*, 149, 106749. <https://doi.org/10.1016/j.cie.2020.106749>
- Issa, M. (2021). Expeditious covid-19 similarity measure tool based on consolidated sca algorithm with mutation and opposition operators. *Applied Soft Computing*, 104, 107197. <https://doi.org/10.1016/j.asoc.2021.107197>
- Jones, D., Snider, C., Nassehi, A., et al. (2020). Characterising the digital twin: A systematic literature review. *CIRP Journal of Manufacturing Science and Technology*, 29, 36–52. <https://doi.org/10.1016/j.cirpj.2020.02.002>
- Karimi-Mamaghan, M., Mohammadi, M., Meyer, P., et al. (2022). Machine learning at the service of meta-heuristics for solving combinatorial optimization problems: a state-of-the-art. *European Journal of Operational Research*, 296(2), 393–422. <https://doi.org/10.1016/j.ejor.2021.04.032>
- Karmouni, H., Chouiekh, M., Motahhir, S., et al. (2022). A fast and accurate sine-cosine mppt algorithm under partial shading with implementation using arduino board. *Cleaner Engineering and Technology*, 9, 100535. <https://doi.org/10.1016/j.clet.2022.100535>
- Kiran, D. (2019). Chapter 10 - forecasting. In D. Kiran (Ed.), *Production Planning and Control* (pp. 141–156). Butterworth-Heinemann. <https://doi.org/10.1016/B978-0-12-818364-9.00010-X>
- Kiyokawa, T., Shirakura, N., Wang, Z., et al. (2023). Difficulty and complexity definitions for assembly task allocation and assignment in human-robot collaborations: A review. *Robotics and Computer-Integrated Manufacturing*, 84, 102598. <https://doi.org/10.1016/j.rcim.2023.102598>
- Kolla, S., Minufekr, M., & Plapper, P. (2019). Deriving essential components of lean and industry 4.0 assessment model for manufacturing smes. *Procedia CIRP*, 81, 753–758. <https://doi.org/10.1016/j.procir.2019.03.189>
- Kuo, T., & Wang, K. J. (2022). A hybrid k-prototypes clustering approach with improved sine-cosine algorithm for mixed-data classification. *Computers & Industrial Engineering*, 169, 108164. <https://doi.org/10.1016/j.cie.2022.108164>
- Langlotz, P., Siedler, C., & Aurich, J. C. (2021). Unification of lean production and industry 4.0. *Procedia CIRP*, 99, 15–20. <https://doi.org/10.1016/j.procir.2021.03.003>. 14th CIRP Conference on Intelligent Computation in Manufacturing Engineering, 15-17 July 2020.

- Lei, K., Guo, P., Wang, Y., et al. (2022). Solve routing problems with a residual edge-graph attention neural network. *Neurocomputing*, 508, 79–98. <https://doi.org/10.1016/j.neucom.2022.08.005>
- Li, S., Fang, H., & Liu, X. (2018). Parameter optimization of support vector regression based on sine cosine algorithm. *Expert Systems with Applications*, 91, 63–77. <https://doi.org/10.1016/j.eswa.2017.08.038>
- Lu, W., Chen, J., Fu, Y., et al. (2023). Digital twin-enabled human-robot collaborative teaming towards sustainable and healthy built environments. *Journal of Cleaner Production*, 412, 137412. <https://doi.org/10.1016/j.jclepro.2023.137412>
- Luo, J., Li, C., Fan, Q., et al. (2022). A graph convolutional encoder and multi-head attention decoder network for tsp via reinforcement learning. *Engineering Applications of Artificial Intelligence*, 112, 104848. <https://doi.org/10.1016/j.engappai.2022.104848>
- Lyu, F., Fan, X., Ding, F., et al. (2021). Prediction of the axial compressive strength of circular concrete-filled steel tube columns using sine cosine algorithm-support vector regression. *Composite Structures*, 273, 114282. <https://doi.org/10.1016/j.compstruct.2021.114282>
- Marinelli, M., Deshmukh, A. A., Janardhanan, M., et al. (2021). Lean manufacturing and industry 4.0 combinative application: Practices and perceived benefits. *IFAC-PapersOnLine*, 54(1), 288–293. <https://doi.org/10.1016/j.ifacol.2021.08.034>. 17th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2021.
- Mirjalili, S. (2016). Sca: A sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems*, 96, 120–133. <https://doi.org/10.1016/j.knsys.2015.12.022>
- Motta, J., Moreno, H., & Ascúa, R. (2019). Industria 4.0 en mipymes manufactureras de la argentina. *Documentos de Proyectos (LC/TS.2019/93)*, Santiago, Comisión Económica para América Latina y el Caribe (CEPAL), https://repositorio.cepal.org/bitstream/handle/11362/45033/1/S1900952_es.pdf
- Nazari, M., Oroojlooy, A., & Snyder, L. V., et al. (2018). Reinforcement learning for solving the vehicle routing problem. [arXiv:1802.04240](https://arxiv.org/abs/1802.04240)
- Ohno, T. (1988). *Toyota Production System: Beyond Large-Scale Production*. UK: Taylor & Francis.
- Otoni, A. L. C., Nepomuceno, E. G., Oliveira, M. S. D., et al. (2022). Reinforcement learning for the traveling salesman problem with refueling. *Complex & Intelligent Systems*, 8, 2001–2015. <https://doi.org/10.1007/s40747-021-00444-4>
- Perez Colo, I., Pirozdo, B., & Saavedra Sueldo, C. (2020). *Simulai*. <https://simulai.readthedocs.io/en/latest/?badge=latest>
- Pizoń, J., & Gola, A. (2023). Human-machine relationship-perspective and future roadmap for industry 5.0 solutions. *Machines*. <https://doi.org/10.3390/machines11020203>
- Saavedra Sueldo, C., Perez Colo, I., De Paula, M., et al. (2022). Ros-based architecture for fast digital twin development of smart manufacturing robotized systems. *Annals of Operations Research*. <https://doi.org/10.1007/s10479-022-04759-4>
- Saavedra Sueldo, C., Villar, S. A., De Paula, M., et al. (2021). Integration of ros and tecnomatix for the development of digital twins based decision-making systems for smart factories. *IEEE Latin America Transactions*, 19(9), 1546–1555. <https://doi.org/10.1109/TLA.2021.9468608>
- Semeraro, C., Lezoche, M., Panetto, H., et al. (2021). Digital twin paradigm: A systematic literature review. *Computers in Industry*, 130, 103469. <https://doi.org/10.1016/j.compind.2021.103469>
- Shah, D., & Patel, P. (2018). Productivity improvement by implementing lean manufacturing tools in manufacturing industry. *International Research Journal of Engineering and Technology (IRJET)*, 5, 3794–3798.
- Singh, S., Singh, A., Kapil, S., et al. (2022). Utilization of a tsp solver for generating non-retractable, direction favouring toolpath for additive manufacturing. *Additive Manufacturing*. <https://doi.org/10.1016/j.addma.2022.103126>
- Skinderowicz, R. (2022). Improving ant colony optimization efficiency for solving large tsp instances. *Applied Soft Computing*, 120, 108653. <https://doi.org/10.1016/j.asoc.2022.108653>
- Stączek, P., Pizoń, J., Danilczuk, W., et al. (2021). A digital twin approach for the improvement of an autonomous mobile robots (amr's) operating environment-a case study. *Sensors*. <https://doi.org/10.3390/s21237830>
- Sutton, R. S., & Barto, A. G. (1998). *Reinforcement learning: An introduction*. USA: MIT Press.
- Thirunavukkarasu, M., Sawle, Y., & Lala, H. (2023). A comprehensive review on optimization of hybrid renewable energy systems using various optimization techniques. *Renewable and Sustainable Energy Reviews*, 176, 113192. <https://doi.org/10.1016/j.rser.2023.113192>
- Uriarte, A. G., Ng, A. H., & Moris, M. U. (2018). Supporting the lean journey with simulation and optimization in the context of industry 4.0. *Procedia Manufacturing*, 25, 586–593. <https://doi.org/10.1016/j.promfg.2018.06.097>. proceedings of the 8th Swedish Production Symposium (SPS 2018).
- Wang, Q., & Tang, C. (2021). Deep reinforcement learning for transportation network combinatorial optimization: A survey. *Knowledge-Based Systems*, 233, 107526. <https://doi.org/10.1016/j.knsys.2021.107526>
- Watkins, C. J. C. H. (1989). *Learning From Delayed Rewards*. PhD thesis, University of Cambridge
- Watkins, C. J. C. H., & Dayan, P. (1992). Q-learning. *Machine Learning*, 8, 279–292. <https://doi.org/10.1007/BF00992698>
- Winston, W. L. (2004). *Operations Research, Applications and Algorithms* (4th ed.). Boston: Brooks/Cole-Thomson Learning.
- Womack, J., Jones, D., & Roos, D. (2007). *The Machine That Changed the World: The Story of Lean Production- Toyota's Secret Weapon in the Global Car Wars That Is Now Revolutionizing World Industry*. USA: Free Press.
- Yamazaki, Y., Shigematsu, K., Kato, S., et al. (2017). Design method of material handling systems for lean automation-integrating equipment for reducing wasted waiting time. *CIRP Annals*, 66(1), 449–452. <https://doi.org/10.1016/j.cirp.2017.04.011>
- Yang, Q. Y., Chu, S. C., Pan, J. S., et al. (2020). Sine cosine algorithm with multigroup and multistrategy for solving cvrp. *Mathematical Problems in Engineering*, 2020, 8184254.
- Yaixin, W., Wen, S., Zhiguang, C., et al. (2022). Learning improvement heuristics for solving routing problems. *IEEE Transactions on Neural Networks and Learning Systems*, 33(9), 5057–5069. <https://doi.org/10.1109/TNNLS.2021.3068828>
- Zhang, Z., & Yang, J. (2022). A discrete cuckoo search algorithm for traveling salesman problem and its application in cutting path optimization. *Computers & Industrial Engineering*, 169, 108157. <https://doi.org/10.1016/j.cie.2022.108157>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.