Engineering Applications of Artificial Intelligence 123 (2023) 106353

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

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

A control system of rail-guided vehicle assisted by transdifferentiation strategy of lower organisms

Yuan-Hao Jiang [a,b], Shang Gao [a,b,*], Yu-Hang Yin [a,b], Zi-Fan Xu [a], Shao-Yong Wang [a]

[a] School of Computer, Jiangsu University of Science and Technology, Zhenjiang, 212100, China
[b] Machine Learning and New Software Technology Institute, Jiangsu University of Science and Technology, Zhenjiang, 212100, China

ARTICLE INFO

Keywords:
Evolutionary algorithm
Large-scale optimization
Rail-guided vehicle
Logistics management
Transdifferentiation

ABSTRACT

Rail-guided vehicle is a logistics management device widely used to perform various material handling operations instead of manual labor. In processing scenarios, the dimensions of the material transfer path of a rail-guided vehicle are typically very large, which makes the optimization of the material transfer path very difficult. The transdifferentiation behavior of lower organisms was introduced into the evolutionary algorithm, and a large-scale differential evolution algorithm based on the transdifferentiation strategy was proposed, for achieving high-efficiency processing. This strategy makes it possible for some individuals with poor fitness to reach maturity again and be selected for the next iteration after losing some information and returning to their juvenile stage, which helps maintain the diversity of the population. Simulation results show that the proposed algorithm not only achieves an average 25.68% higher output rate than the comparison algorithms on the test cases but also has an excellent and stable effect distribution level on the extended problem space, which shows that the superiority of the proposed algorithm is not affected by the processing parameters. This research is expected to provide technical guidance for the processing of key components in the ship and aviation manufacturing industries. The code with a 31-page manual is available on our project homepage https://github.com/MLNST-JUST/DE-TS.

1. Introduction

At present, the rapid development of artificial intelligence has played a significant role in the growth of the world economy. At the same time, it has also achieved breakthrough research results in many fields such as face recognition (Shao et al., 2018), DNA–protein binding recognition (Yin et al., 2022), and early warning of gas concentration (Cai et al., 2021). It also provides effective solutions for many major engineering problems. The metaheuristic algorithm is a type of artificial intelligence algorithm with extensive applicability (Loubière et al., 2018; Muhammad et al., 2018). As an important part of the field of computational intelligence, it has achieved outstanding performance in many tasks such as servo system control (Pozna et al., 2022; Precup et al., 2021; Zamfirache et al., 2022), polynomial control design (Madiouni et al., 2019), and time series forecasting (Abdulkarim and Engelbrecht, 2021). At present, metaheuristic algorithms have been effectively applied in many major projects and have been applied to key processes such as mechanical control, logistics control, processing control, and other key processes.

With the rapid development in the fields of digital twin and mechatronics, integration (Ogunsina and DeLaurentis, 2022), unmanned (Bhaskara et al., 2021), and intelligence (Mohamadi et al., 2022; Zhang et al., 2022) have become the main development directions of production and processing systems. To achieve this goal, rail-guided vehicle (RGV) is widely used in various production and processing systems such as aviation safety (Kim et al., 2018), parts processing (Wang et al., 2014), magnetic levitation control (Cho and Kim, 2014), and intelligent storage (Kou et al., 2018), which helps to replace human execution of material loading and unloading, material handling, material cleaning, and other material handling operations. Meanwhile, This also helps to achieve finer control of the material flow state (Zhang and Wang, 2019). This not only reduces the instability of the material flow process but provides a base for the further development of digital twin technology as well (Lee et al., 2021).

In recent years, the use of RGV equipment for integrated scheduling control of high-precision computer number controllers (CNC) has become very common (Fan et al., 2019; Zhang and Wang, 2019). In this operation scenario, the RGV equipment performs integrated control on the CNC, so that the CNC can stably and efficiently complete the scheduled processing tasks under appropriate material supply conditions. This practical problem can be called the RGV dynamic scheduling problem (RDSP). Unlike the classic workshop scheduling problems such

* Corresponding author.
E-mail addresses: jiangyuanhao@stu.just.edu.cn (Y.-H. Jiang), gao_shang@just.edu.cn (S. Gao), yyh@stu.just.edu.cn (Y.-H. Yin), 202070006@stu.just.edu.cn (Z.-F. Xu), sy_wang@stu.just.edu.cn (S.-Y. Wang).

https://doi.org/10.1016/j.engappai.2023.106353
Received 24 July 2022; Received in revised form 6 March 2023; Accepted 19 April 2023
Available online xxxx
0952-1976/

as job-shop scheduling problem (JSP) (Applegate and Cook, 1991), dynamic job-shop scheduling problem (DJSP) (Jain and Meeran, 1999), flexible job-shop scheduling problem (FJSP) (Pezzella et al., 2008), dynamic flexible job-shop scheduling problem (DFJSP) (Zhang et al., 2021a,b) and distributed job-shop scheduling problem (DSSP) (Wang and Peng, 2020), the RDSP not only requires finding a feasible and efficient processing sequence for each processing equipment but also needs to consider thoroughly how to use RGV equipment for real-time material supply. This research helps keep the material flow capability of the processing system matched to its processing capacity. In general, the research on the RDSP additionally considers the material supply and circulation process and transforms the workshop processing problem into the RGV scheduling control problem, which is more closely related to the real processing scene.

At present, many important works have emerged on the RDSP. In the workshop scene of RGV, Fan et al. (2019) designed an improved genetic algorithm for solving the RDSP (IGAR) based on the cyclic perturbation coding method — its algorithm performance is significantly better than the traditional genetic algorithm. Dotoli and Fanti (2005) constructed a colored Petri net model in the RGV workshop. Martina et al. (2018) studied the relationship between the throughput of the processing system and the number of RGVs in the system and pointed out the nonlinear relationship between the two. Lee et al. (1996) adopted the first in first out (FIFO) strategy and designed a method to determine the optimal number of RGV installations in the application scenario of a warehouse circular transportation system. Xiao (2019) added the Kraskar selection operator to the greedy algorithm and designed a dynamic scheduling simulation model for RGV material processing, which provided a feasible solution for the path selection between CNCs. Wang et al. (2018) proposed a dynamic scheduling simulation model based on the greedy strategy, which realized the local scheduling strategy optimization of RGV. To improve the objectivity and accuracy of the multi-objective weight calculation in the intelligent processing system model, Li (2019) used the entropy weight method and the expert sorting method to calculate two sets of weights, and established a feasible intelligent scheduling model for RDSP. Ding et al. (2020) designed a forward-looking step-by-step model for optimizing the RGV scheduling scheme, introduced a chaotic particle swarm optimization algorithm into the model, and proposed a hybrid method integrating a multi-step processing mechanism and evolutionary algorithm. All of which have made a significant contribution to the research and the development of the RDSP.

In our past work, we carried out a comparative analysis of the metaheuristic algorithms for solving optimization problems (Muhammad et al., 2018), and discussed the application of the proposed hybrid algorithm combining ant colony optimization algorithm with genetic algorithm (Shang et al., 2007a), and the application of the wading across stream algorithm (Gao et al., 2014) on the issue of the traveling salesman problem. In these tasks, we have tested the performance of the metaheuristic algorithms in path searching. We also discussed the application of the proposed improved ant algorithm (Shang, 2008) and Immune genetic algorithm (Shang et al., 2007b) on the weapon–target assignment problem to test the performance of the metaheuristic algorithms in the target matching. Further, based on cost effectiveness analysis, we have designed a system reliability optimization model, mainly including the selection of components and system architecture. In this paper, we mainly studied the RGV scheduling control and CNC processing arrangements under the RDSP. Based on the above discussion, it is not difficult to find that when solving the RDSP, preset rule methods such as the FIFO scheduling strategy or heuristic search method are generally used, or a combination of the two. Since the continuous processing time in the RGV workshop is often very long, the RDSP series of problems are tremendous — the dimension of the decision vector may exceed 50,000. Therefore, the solution to the RDSP is very difficult to search for directly. To this end, this paper proposes a differential evolution algorithm based on the transdifferentiation strategy (DE-TS). The algorithm includes an RGV material transfer sequence generation mechanism based on a debilitating factor, to accelerate the convergence speed in the early stage of the algorithm without compressing the solution space. At the same time, a novel transdifferentiation strategy acts on the evolution process of the population, which enables the population diversity to be effectively maintained. The main contribution of this work is as follows:

(1) On the RDSP, the larger problem scale makes the optimization of the population particularly slow. An effective solution is to achieve dimension reduction operations by setting a loop in each individual in the population. For example, each CNC is required to be accessed only once in the cycle, which is an effective way to rapidly improve the production efficiency of the processing system (Fan et al., 2019). However, on the one hand, such a process of treatment has abandoned a large part of the solution space, which makes many efficient processing strategies may be directly excluded from the search area. On the other hand, although each CNC has the same number of processing times when the processing time of processing required for different processes is large, it may lead to a large processing load gap between different CNCs. This also reduces the upper limit of processing efficiency to a certain extent. To the contradiction between search efficiency and CNC load, the optimal operation of offspring individuals based on debilitating factor was designed. During the iterative process, each individual has a certain probability to set a processing cycle. At this time, the proposed algorithm is distributed according to the generated probability, and the corresponding cycle length is selected for each individual. In the cycle, according to the processing requirements of the current material, the number of allowable access to each CNC will be determined in real-time, so that the difference in the processing load between different CNCs is as small as possible.

(2) Collaborative evolutionary optimization is an evolutionary strategy based on the idea of divide and conquer. To improve the performance in solution searching, the coordinated evolution strategy was introduced in the proposed algorithm. During each iteration, the generated offspring population will be divided into three sub-population according to their fitness from high to low: the surviving population, the injured population, and the eliminated population. On the one hand, the surviving population is responsible for maintaining the superiority of individuals, and those individuals included in it will be sent unconditionally into the next iteration. On the other hand, the injured population is responsible for maintaining the diversity of populations. Each individual in it is partly "damaged" according to its fitness, and its decision variable is partly lost according to the degree of damage. Then it is supplemented by the sampling method. This strategy enables the proposed algorithm to better balance the superiority of individuals and the diversity of the population.

(3) In order to balance CNCs' material processing capabilities and RGV's logistics management capacity, a learning mechanism is designed in the proposed algorithm, which enables individuals to have the ability to learn the distribution information of CNC processing tools from the parent population. This study also gives a calculation method for the mathematical expectation dimension of the RGV material transfer sequence, to prevent potential problems caused by the mismatch between the preset sequence length — on the one hand, the algorithm will terminate prematurely when the sequence is too short. On the other hand, computing resources will be wasted when the sequence is too long. In addition, this research provides the source code for the RDSP series of test problems and the DE-TS algorithm. The source code and its guidance manual of this study are available now on github.

In Section 2, we introduce the test problems used in the experiments and the performance metrics used to measure the obtained solutions. Subsequently, the main framework and details of the DE-TS algorithm are introduced in detail in Section 3. Section 4 describes the experimental setup and comparative results. Finally, conclusions are drawn and future work is outlined in Section 5.

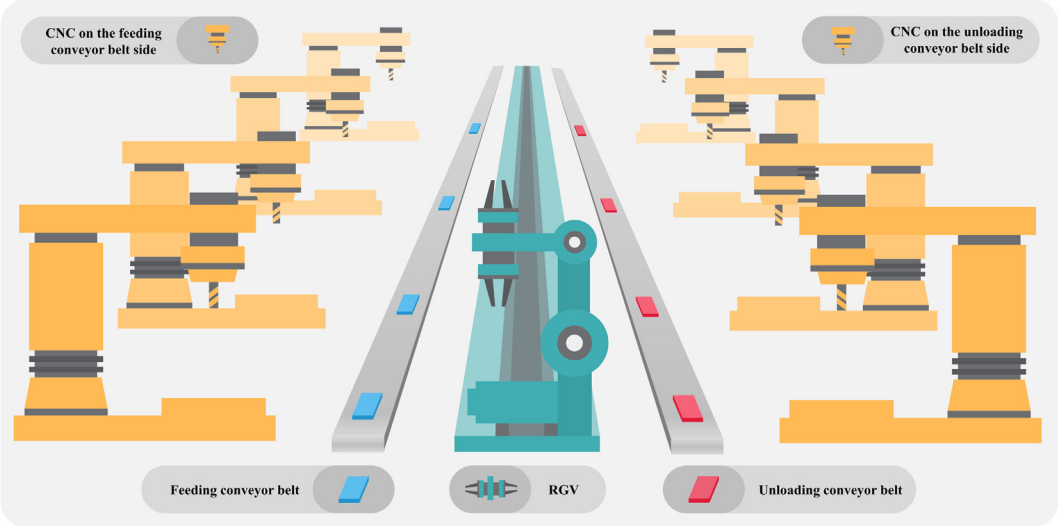

Fig. 1. Schematic diagram of the processing system. Among them, the RGV and its track are in the middle. The loading conveyor belt and the unloading conveyor belt are located on both sides of the RGV track. CNCs are evenly distributed on both sides of the RGV track.

2. RDSP and its evaluation indicator

2.1. Processing system composition

In the RDSP, the production workshop may contain multiple sub-processing systems, each sub-processing system consists of an RGV and its moving track, a set of feeding conveyor belt, a set of unloading conveyor belt, multiple CNCs and their related accessories. Among them, the RGV is composed of mechanical grippers, a clinker cleaning tank, double-headed mechanical arms and other components, which can accurately perform material processing operations such as movement, loading and unloading, material handling, and finished material cleaning according to system instructions. On the one hand, the loading conveyor belt is used to ensure that no matter where the RGV stops, there is raw material available. On the other hand, the unloading conveyor belt is used to ensure that no matter where the RGV stops, the finished material can be placed onto it. Then, the finished material will be sent off the processing system automatically. The CNCs in the processing system are evenly distributed on both sides of the RGV track. Generally, the number of CNCs in each sub-processing system is not less than 8. Depending on the type of tool installed, the CNC can perform the corresponding machining operation on the material. At the same time, each CNC can only perform preset processing operation on one material. Fig. 1 shows a schematic diagram of a machining system with 8 CNCs. It should be noted that the number of CNCs may be different in different machining systems.

2.2. Material processing flow

In the processing system, RGV needs to provide a stable supply of materials for each CNC, put the processed materials into the clinker cleaning tank for cleaning, and finally put them on the unloading conveyor belt to send them off. At the same time, the CNC needs to perform corresponding processing operations on the materials placed by the RGV on its processing table according to the type of tools installed.

Fig. 2 shows the general material processing flow. Firstly, the RGV needs to move in front of a Class I CNC for the first process, and grab a piece of raw material on the feeding conveyor belt with the A-side gripper of the double-headed robotic arm. Then, the double-headed robotic arm will turn 180°, and use the vacant B-side gripper to grab the semi-clinker that has been processed on the Class I CNC machining table. In the next step, the double-headed robotic arm will turn 180° again, and the material gripped by the A-side gripper will be automatically placed on the processing table. Next, the RGV will move to a Class II CNC for the second process, and use the A-side mechanical gripper to hold the processed clinker on the processing table, and immediately turn it over 180°, and then turn the B-side mechanical claw. The semi-clinker gripped by the claws is placed on it. Immediately afterward, the double-headed manipulator arm will turn 180° again, and the clinker held by the mechanical claw on the A-side will be placed in the clinker cleaning tank for cleaning. Finally, the finished material will be placed on the unloading conveyor belt by the RGV to leave the processing system. In Fig. 2, raw material, semi-clinker, clinker and finished material are marked with blue, green, orange and red respectively to distinguish.

Noticeably, in those steps marked in orange in Fig. 2, after the material is placed on the CNC machining table, the CNC will automatically perform the machining operation. The machining operation of CNC is asynchronous with the material transfer operation of RGV. In a nutshell, RGV can leave after completing the feeding operation without waiting for the current CNC to complete the machining operation. It should be noted that some of the steps included in Fig. 2 can be skipped under certain circumstances. For example, when the machining system is first started, all CNC machining tables are empty. When there is no material to be unloaded on the CNC accessed by RGV, RGV will perform the loading operation normally after skipping the unloading operation. At this time, if the currently accessed CNC is type II CNC, the material cleaning step will also be skipped together. In addition, when the machining system is about to stop, all CNC machining tables are gradually emptied. At this time, the RGV will skip the next feeding operation after the normal unloading operation.

2.3. Definition of the optimization problem

In different processing scenarios, the maximum material output value of the RDSP is inconsistent. On the one hand, some studies on the RDSP directly use the output of finished materials as a comparison index (Ding et al., 2020; Fan et al., 2019), which effectively realizes the longitudinal comparison between different algorithms in the same scenario. However, under different processing parameters, such as the number of CNCs or the material transferring time of RGV, it is hard to

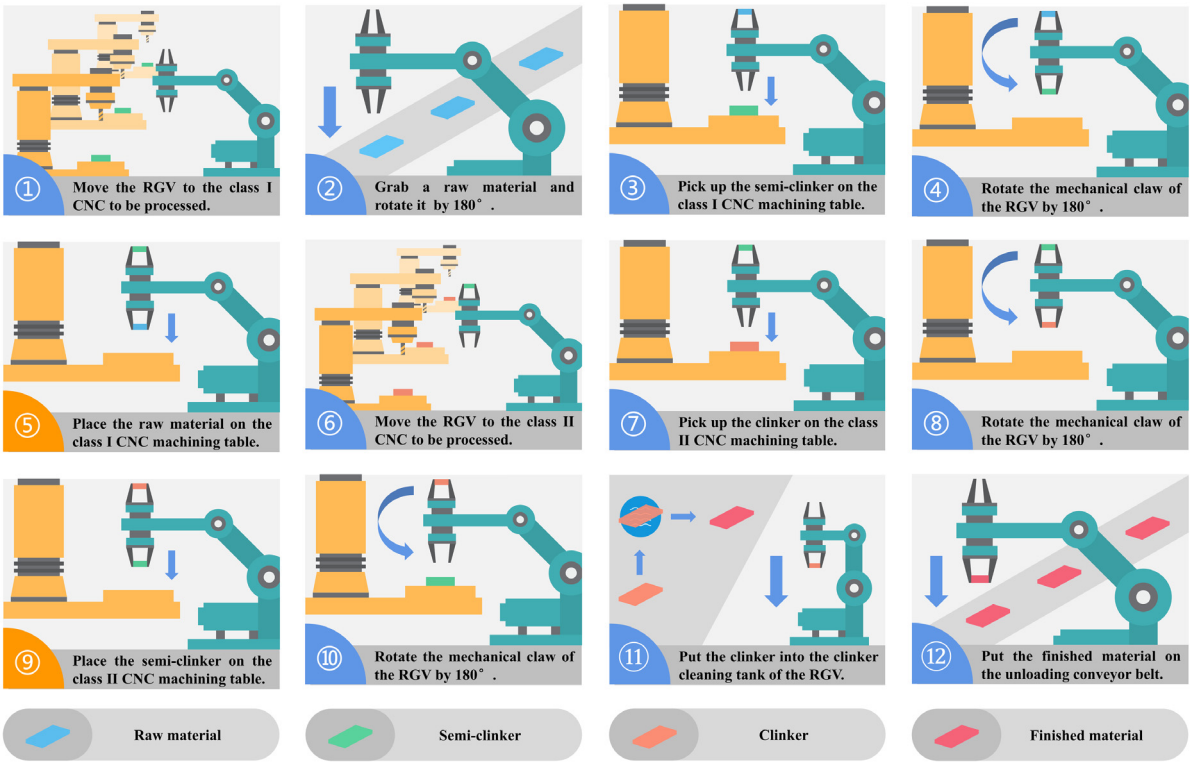

Fig. 2. Material processing flow chart. This figure includes the complete processing process of a raw material being processed into a finished material.

achieve a horizontal comparison of the same algorithm in different scenarios. For example, an algorithm achieves L_1 and L_2 finished material output in Scenario 1 and Scenario 2, respectively. However, these data cannot be used to compare the pros and cons of this algorithm in these two scenarios in that the finished material output does not rule out the influence of different processing parameters on the problem itself. On the other hand, some studies use the total idle time of all CNCs as an evaluation index (Wang et al., 2019). In this way, the utilization rate of the system can be obtained by dividing the total idle time of CNCs by the total processing time. This evaluation indicator provides a solution to horizontal comparison in different scenarios. However, since the load capacity of CNC is greatly affected by the parameters in specific processing scenarios, the system parameters in different processing scenarios may be quite different. In order to enhance the fairness of the experiment, this study designs the loss as an evaluation indicator to evaluate the algorithm performance on the RDSP problem as fairly as possible.

Assuming that all CNCs in the processing system are continuously performing the work cycle of "unloading and processing materials", all CNCs are in an ideal state of full load at this time. Then, the upper limit of the system output $L_{\max}$ is:

$$L_{\max} = \min\left(\sum_{i\in P}\left\lfloor\frac{T}{t_{hi}+t_{li}}\right\rfloor, \sum_{i\in Q}\left\lfloor\frac{T}{t_{hi}+t_{li}}\right\rfloor\right). \tag{1}$$

Among (1), T represents the continuous operation time of the processing system, t_{hi} represents the time required for the CNC i to complete one machining operation, and t_{li} represents the time required for the RGV to complete a loading and unloading operation for the CNC i. Since the loading and unloading operation time is affected by the CNC position, the time spent by the CNC on the loading conveyor belt side and the unloading conveyor side belt in this step is slightly different. The CNC equipped with the tools required for the first machining process is called the class I CNC, and the set of all the class I CNCs is P; The CNC equipped with the tools required for the second machining process is called the class II CNC, and the set of all the class II CNCs is Q, then there are:

$$\begin{cases} U_{\text{CNC}} = \{P, Q\} \\ s.t.\ P \cup Q = \varnothing. \end{cases} \tag{2}$$

Among (2), U_{CNC} represents the universal set of all CNCs. The intersection of P and Q is an empty set, indicating that each CNC can only install a type of processing tool during a processing process. At this time, the upper limit of the system is limited by the processing capacity of different types of CNC. After obtaining the upper limit of the system output $L_{\max}$, the result of the optimization problem can be calculated by (3). During a processing process, assuming the number of finished materials produced under the scheduling scheme generated by an algorithm is L_{real}, saying that *loss* is the processing efficiency loss of the processing system, the optimization problem can be defined as:

$$\begin{cases} \min \quad loss(DV) = 1 - F(DV) \\ s.t. \quad F(DV) = \dfrac{L_{\text{real}}}{L_{\max}} \times 100\% \\ \qquad DV = \{DV_{\text{CNC}}, DV_{\text{RGV}}\} \\ \qquad L_{\text{real}} = G(DV_{\text{CNC}}, DV_{\text{RGV}}) \\ \qquad DV_{\text{CNC}}(i) = \begin{cases} 0, & \text{CNC}_i \in P \\ 1, & \text{CNC}_i \in Q \end{cases}, i = 1, 2, \ldots, N_{\text{CNC}} \\ \qquad DV_{\text{RGV}}(j) = 100 * TN_{\text{I}}(j) + TN_{\text{II}}(j), \\ \qquad\qquad j = 1, 2, \ldots, E(D) - N_{\text{CNC}} \\ \qquad TN_{\text{I}}(k) \in P,\ k = 1, 2, \ldots, E(D) - N_{\text{CNC}} \\ \qquad TN_{\text{II}}(l) \in Q,\ l = 1, 2, \ldots, E(D) - N_{\text{CNC}}. \end{cases} \tag{3}$$

Among (3), F represents the output rate, and its value corresponds to the individual's fitness level. DV is the decision variable, which is

composed of the decision variable DV_{CNC} in the CNC encoding part, and the decision variable DV_{RGV} in the RGV encoding part. In addition, G is the calculation function of the actual number of finished materials produced. $L_{\mathrm{real}} = G(DV_{\mathrm{CNC}}, DV_{\mathrm{RGV}})$ indicates that the tool installation is performed according to DV_{CNC}, and the material transfer control is performed according to DV_{RGV}, and the actual number of finished materials that are produced is L_{real}. On the one hand, the DV_{CNC} section is encoded by 0 and 1, and its length is consistent with the number of CNCs in the processing system. Each value in DV_{CNC} corresponds to the type of a CNC. On the other hand, DV_{RGV} uses four digits to store the transfer path of each material. $DV_{\mathrm{RGV}}(j)$ is used to indicate the transfer path of the jth material. The first two digits TN_{I} and the latter two digits TN_{II} are used to indicate the number of type I CNC and type II CNC accessed by the material. The length of the decision variable in this part is $E(D) - N_{\mathrm{CNC}}$. $E(D)$ is the length of the decision variable DV, indicating the mathematical expectation value of the individual dimension D. Its calculation method will be introduced in detail in the (4) in 3.2 section. It can be seen that the tool distribution of the processing system and various processing parameters jointly determine the upper limit of the system output $L_{\max}$ of the production workshop, and the decision vectors generated by different algorithms determine the minimum *loss* that the processing system can achieve under the limitation of the upper limit of $L_{\max}$. The lower the *loss* value of the system, the better the corresponding algorithm.

3. The proposed algorithm

3.1. The structure of the proposed algorithm

Fig. 3 shows the structure of the DE-TS and its differences from the DE (Differential Evolution) (Storn and Price, 1997). Among them, the core steps of the DE are marked with a gray area, which mainly includes three parts: mutant operation, cross operation and selection operation. The blue area shows the strategy introduced by the DE-TS algorithm. During the execution of the proposed algorithm, after performing the mutation and cross operation of DE, the original selection operation will no longer be performed, and it is replaced by the individual optimization and transdifferentiation strategy in DE-TS. At this time, each individual will perform an optimal operation based on debilitating factor, so that the distribution of decision variables of the offspring population is more matched with the needs of real processing scenarios. Subsequently, the optimized offspring population will merge with the parent population. The merging population will be sorted according to the fitness, and from high to low, it is divided into the surviving population, the injured population, and the eliminated population. At this time, the injured population will lose a part of its decision variable based on its fitness. The lost part will be made up by executing the transdifferentiation strategy. Then, the injured population will be merged with the surviving population into the next iteration. The detail of it will be introduced in the subsequent part of this chapter. Every time the *loss* is calculated, the number of evaluations *FE* is added with 1. When *FE* reaches the max*FE*, the algorithm ends.

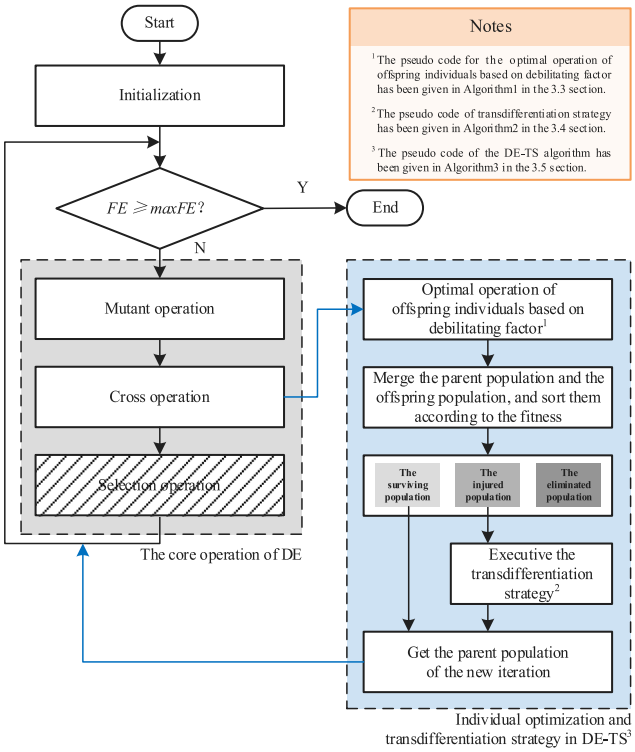

Fig. 3. The structural diagram of the DE-TS. The strategy introduced by the proposed algorithm based on the DE is marked in the blue area. The *FE* and max*FE* in this figure refer to the current number of function evaluations and the maximum number of function evaluations, respectively.

3.2. Population initialization

When solving this problem, the first step is to assign appropriate machining tools to all CNCs, and the next step is to generate a suitable material transfer path for the RGV. Affected by the number of CNCs and the continuous processing time, the material transfer path of RGV may be very long — its length is generally between 30,000 and 50,000 dimensions, which adds an obstacle to the optimization of the problem. During the population initialization, the initial population will be randomly generated in the effective solution space.

Each individual can be represented by a one-dimensional vector, as Fig. 4 shows an example of coding for an individual. On the one hand, when there are N_{CNC} CNCs in the machining system, the front N_{CNC} dimension of the decision vector is used to store the CNC tool information - 1 represents class I CNC, and 0 represents class II CNC. On the other hand, the remainder of the decision vector is used to store the material transfer path of the RGV, with a 4-digit number per dimension to store one complete machining as shown in Fig. 2.

Among them, the first two digits and the last two digits represent the relative numbers of the type I CNC and the type II CNC visited in this machining process, respectively. The relative number is obtained by classifying the original CNC number according to the tool category, and then renumbering it from small to large within each category. Using relative numbers instead of absolute numbers can facilitate the repair of out-of-bounds individuals. For instance, there are only 7 CNCs of class I in a machining system. However, when the algorithm repairs the material transfer path of the RGV of a certain individual I, it finds a class I CNC with a relative number greater than 7. In this case, it can be swiftly and briskly determined that the current individual I is out of bounds. Subsequently, the out-of-bounds part of I will be quickly mapped back into the effective solution space.

Compared to absolute numbering, relative numbering avoids a large number of query operations required to fix the individual and speeds up processing. In a population, all individuals have the same dimension. The expectation of the individual dimension can be calculated by (4):

$$\begin{cases} E(D) = n_{\mathrm{CNC}} + \max(D_{\mathrm{I}}, D_{\mathrm{II}}) \\ s.t.\ D_{\mathrm{I}} = \left\lfloor \dfrac{T}{t_h^{(\mathrm{I})}} \right\rfloor \left\lceil \dfrac{t_h^{(\mathrm{I})} n_{\mathrm{CNC}}}{t_h^{(\mathrm{I})} + t_h^{(\mathrm{II})}} \right\rceil \\ D_{\mathrm{II}} = \left\lfloor \dfrac{T}{t_h^{(\mathrm{II})}} \right\rfloor \left\lceil \dfrac{t_h^{(\mathrm{II})} n_{\mathrm{CNC}}}{t_h^{(\mathrm{I})} + t_h^{(\mathrm{II})}} \right\rceil . \end{cases} \tag{4}$$

Among (4), $E(D)$ is the expected value of the individual dimension D, $\max(D_{\mathrm{I}}, D_{\mathrm{II}})$ is the upper limit value of the material transfer path

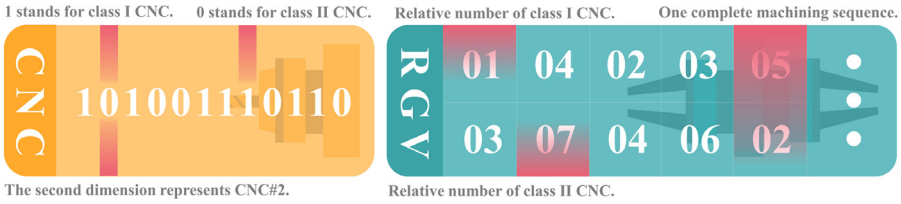

Fig. 4. The individual coding graph consists of two parts. The former is the CNC class code, and the latter is the RGV material transfer path code.

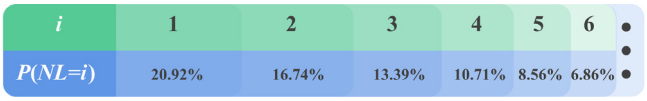

Fig. 5. Probability distribution map of NL when $\mu = 0.8$.

dimension of the RGV, D_{I} and D_{II} are the upper limit values of the dimension considering only the first process or the second process, respectively. $t_h^{(\mathrm{I})}$ and $t_h^{(\mathrm{II})}$ are the time required to complete a material processing for class I CNC and class II CNC, and these values may vary with processing requirements. Taking process one as an example, The results of rounding down $T/t_h^{(\mathrm{I})}$ represents the maximum number of repeated machining times of class I CNC under ideal conditions within the continuous operation duration T.

3.3. Generation of offspring

After the population is initialized, the parent generation will reproduce by the standard DE algorithm (Storn and Price, 1997), and the generated offspring are still in their juvenile stage. After the out-of-bounds solution is automatically repaired, each offspring will gradually develop from the initial juvenile stage and enter the maturity stage. Since RDSP is a large-scale problem, in order to speed up the convergence speed in the early stage of algorithm operation, this paper gives the concept of the atomic work cycle, and calculates a suitable cycle length for each offspring — this cycle length is an integer multiple of the atomic work cycle, and this multiple is called cycle number NL. NL is determined according to the parameter of the debilitating factor μ. The weight in the case of $NL = 1$ is assigned a value of 1. Subsequently, each time NL is increased by one, and its weight is multiplied by μ. Finally, assign probabilities to different values by weight. Fig. 5 shows the probability distribution map of NL when $\mu = 0.8$.

In the machining system, the RGV will visit a class I CNC and a class II CNC, in turn, before it completes one machining process each time. Therefore, although the machining time and the number may be dissimilar for different classes of CNC, the total number of visits is the same for each class of CNC. This imbalance is further exacerbated if each CNC is required to be visited the same number of times during an atomic duty cycle. To deal with this contradiction, this paper defines the atomic work cycle as a sequence of length $2Tool^{(\mathrm{I})}{\cdot}Tool^{(\mathrm{II})}$, where $Tool^{(\mathrm{I})}$ and $Tool^{(\mathrm{II})}$ are the number of CNCs installed with class I machining tools and class II machining tools, respectively. In an atomic work cycle, each class I CNC is visited $Tool^{(\mathrm{II})}$ times, and each type II CNC is visited $Tool^{(\mathrm{I})}$ times, for a total of $2Tool^{(\mathrm{I})}{\cdot}Tool^{(\mathrm{II})}$ times. At this time, it is not only guaranteed that the total number of visits of each class of CNC is the same, but also that the load of the same class of CNC is close to the same. In each atomic work cycle, $Tool^{(\mathrm{I})}{\cdot}Tool^{(\mathrm{II})}$ processing can be completed and the corresponding amount of finished material can be produced.

Algorithm 1: Optimal operation of offspring individuals based on debilitating factor

Input : Debilitating factor μ, individual I
Output: The optimized individual I
1: **Initialization**: Perform the decoding operation on the individual I, and get the CNC tool installation list *ToolList* and the RGV material transfer path R_I;
2: Repair the CNC tool installation list *ToolList* of individual I, and obtain the number of classes I and II CNCs after repairing $Tool^{(\mathrm{I})}$ and $Tool^{(\mathrm{II})}$;
3: Calculate the length AWC_I of the atomic duty cycle of individual I: $AWC_I \leftarrow 2Tool^{(\mathrm{I})} \cdot Tool^{(\mathrm{II})}$;
4: Construct a monotonically decreasing proportional sequence a with μ as a common ratio, such that $sum(a) \leftarrow 1$;
5: According to $P(NL_I{=}{=}i) \leftarrow a(i)$, determine the number of atomic work cycles NL_I included in the initial RGV material transfer path IR_I of individual I;
6: $length(IR_I) \leftarrow AWC_I \cdot NL_I$;
7: $IR_I \leftarrow R_I(1{:}\ min(length(IR_I), E(D){-}N_{CNC}))$;
8: Repair the IR_I in random order, so that the number of visits to each class I or class II CNC is $AWC_I \cdot Tool^{(\mathrm{II})}$ and $AWC_I \cdot Tool^{(\mathrm{I})}$, respectively;
9: **if** $length(IR_I) < E(D){-}N_{CNC}$
10: | **while** $length(IR_I) < E(D){-}N_{CNC}$ **do**
11: | | $IR_I \leftarrow [IR_I, IR_I]$
12: | **end**
13: **end**
14: $R_I \leftarrow IR_I(1{:}\ E(D){-}N_{CNC})$
15: Using *ToolList* and R_I to re-encode and form the optimized individual I;
16: **return** I

For a certain individual I, after the value of NL_I is determined, the appropriate cycle length can be determined for I according to the definition of the atomic work cycle. At this time, the first $2NL_I{\cdot}Tool^{(\mathrm{I})}{\cdot}Tool^{(\mathrm{II})}$ bits of the RGV material transfer path of the current individual I will be extracted, and the repair operation will be performed in random order. This operation allows each class I CNC to be visited $NL_I{\cdot}Tool^{(\mathrm{II})}$ times, while each class II CNC is visited $NL_I{\cdot}Tool^{(\mathrm{I})}$ times. If the length of the final generated loop is insufficient, the loop is repeated until the dimension requirement is met. In fact, if the process is directly disassembled into fixed-length loops, the convergence speed of the algorithm in the early stage will be accelerated to a certain extent. However, this also greatly compresses the solution space, potentially discarding the optimal solution. At this time, it is more effective to decide whether to execute the cycle processing according to the length of the RGV material transfer path after selecting the value of NL_I and the weakening factor μ. Successively, it can prevent the solution space from being greatly reduced and speed up the convergence speed of the algorithm to a certain extent.

Specifically, after the offspring population is generated according to the DE, each individual performs an individual optimization according to the pseudocode shown in Algorithm 1. The following is further described with reference to the line number given in Algorithm 1. First, the decoding operation is performed on the individual I to be optimized, to obtain the CNC tool installation list *ToolList* and the RGV material transfer path R_I. This step corresponds to the first line in Algorithm 1. Next, the ToolList of the individual I will be automatically repaired to prevent the number of tools required for machining in different processes from being too unbalanced. This step corresponds to the second line in Algorithm 1. Subsequently, the length AWC_I of one

atomic work cycle of I will be automatically calculated, and the initial RGV material transfer path IR_I of I will be composed of NL_I atomic work cycles, where NL_I is obtained by probability based on a given debilitating factor. This part is corresponding to lines 3–7 in Algorithm 1. Calculation related to the debilitating factor has been introduced in the previous two paragraphs. Subsequently, as shown in the 8th line of Algorithm 1, fix the initial RGV material transfer path IR_I in a random order, so that the load of different types of CNC is as close as possible. When the dimension of the IR_I is longer than the expected encoding length, only the first $E(D)$–N_{CNC} bits are truncated as the RGV material transfer path R_I. Otherwise, a circular stuffing operation is performed on it, resulting in an R_I of dimension $E(D)$–N_{CNC} bits. This part corresponds to lines 9–14 in Algorithm 1. According to line 15 in Algorithm 1, *ToolList* and R_I will be re-encoded to form the optimized individual I.

Algorithm 2: Transdifferentiation strategy

Input : Individual I
Output: Individual I after performing the transdifferentiation strategy
1: **Initialization**: Perform the decoding operation on the individual I, and get the CNC tool installation list *ToolList* and the RGV material transfer path R_I;
2: I. *ToolList*. Disorder( );
3: I. R_I. Delete( );
// Individual I returns to infancy: *ToolList* is disrupted and the information of R_I is lost.
4: Obtain the number of class I and class II CNCs after restoration $Tool^{(I)}$ and $Tool^{(II)}$;
5: Individual I re-enters maturity stage: Using the method of offspring individual optimization based on the debilitating factor, one atomic work cycle of individual I is obtained and used as the initial RGV material transfer path IR_I;
6: **if** $length(IR_I) < E(D)$-N_{CNC}
7: | **while** $length(IR_I) < E(D)$-N_{CNC} **do**
8: | | $IR_I \leftarrow [IR_I, IR_I]$
9: | **end**
10: **end**
11: $R_I \leftarrow IR_I(1: E(D)$-$N_{CNC})$
12: Using *ToolList* and R_I to recode and form individual I after executing the transdifferentiation strategy;
13: **return** I

3.4. Population update based on the transdifferentiation strategy

In the standard DE algorithm (Storn and Price, 1997), the offspring population generated in each iteration will be mixed with the parent population. Then, the mixed population will select individuals to enter the next iteration according to the greedy strategy. In nature, whenever individuals face a variety of dangers such as natural disasters, predation dangers, etc., the situation is not always polarized — some individuals die instantly while others remain unscathed; in reality, many individuals do not die despite being seriously injured. Next, these individuals will begin to heal themselves—especially lower creatures (Wang et al., 2021). Among them, some organisms will retreat from the maturity stage to the juvenile stage in the process of self-healing after being severely injured, such as the turritopsis nutricula (Piraino et al., 1996). This phenomenon is called transdifferentiation (Eguchi and Kodama, 1993). Many studies have reported that the introduction of the population cooperation mechanism into evolutionary algorithms can effectively improve the algorithm effect (Blot and Petke, 2021; Chen et al., 2021; Siddique et al., 2021; Tang et al., 2021). In the proposed DE-TS algorithm, some individuals involved in the transdifferentiation process undertake the task of maintaining population diversity. At the same time, some individuals with the highest fitness undertake the task of maintaining the fitness level of the population. These populations can effectively improve the algorithm effect through cooperation.

The proposed algorithm treats each iteration as a natural selection for population update operations in the mixed parent and offspring populations. When natural selection occurs, half of the population with lower fitness die initially—those individuals who have been harmed beyond the limit of their ability to heal themselves. Among the surviving individuals, some of them with higher fitness have suffered less damage, while the remaining individuals have suffered greater trauma in the disaster, but are still within the ability to recover — the proportion of this part of the population is related to environmental stress α. At this point, these individuals will immediately enter the transdifferentiation stage, and in the process of self-healing, return from the maturity stage to the juvenile stage.

Algorithm 3: Framework of the Proposed Algorithm

Input : The number of CNCs N_{CNC}, the continuous working time T_{max} of the processing equipment, the material processing parameter list of RGV and the processing time of each step of the material
Output: The obtained best individual I_{best} and the corresponding system *loss*
1: **Initialization**: Randomly initialize the population
2: Calculate the tool allocation ratio according to the processing time of each step of the material;
3: $FE \leftarrow 0$
4: **while** $FE < maxFE$ **do**
5: | The parent population *ParPop* generates the offspring population *OffPop* through the standard DE algorithm;
6: | For each individual in the offspring population *OffPop*, perform the offspring individual optimization based on the debilitating factor;
7: | *Population* = sort([*ParPop*, *OffPop*]);
| // After the parent and offspring populations are merged, they are sorted by fitness.
8: | $FE \leftarrow FE + PopSize$;
| // During this sorting process, the offspring population requested the *PopSize* function evaluation.
9: | **for** i=1 to 2*PopSize* **do**
10: | | **if** $i < PopSize$ **then**
11: | | | Half of the individuals with lower fitness are discarded;
12: | | **elseif** $(1\text{-}\alpha) \cdot PopSize < i < PopSize$ **then**
13: | | | $NewPop \leftarrow [NewPop, Population(i).\text{TS}()]$;
| | | // $\alpha \cdot PopSize$ individuals with moderate fitness execute the transdifferentiation strategy, where α is the environmental pressure.
14: | | | $FE \leftarrow FE + 1$;
| | | // Each new individual requests a function evaluation.
15: | | **else**
16: | | | $NewPop \leftarrow [NewPop, Population(i)]$;
| | | // The remaining individuals with better fitness are directly saved into the new population.
17: | | **end**
18: | **end**
19: | *ParPop* $\leftarrow$ *NewPop*;
| // Use *NewPop* as the parent population for the next iteration.
20: **end**
21: Find the best individual I_{best} and the corresponding *loss* generated in the iterative process;
22: **return** I_{best} and its *loss*

The pseudocode of the transdifferentiation strategy is shown in Algorithm 2. First of all, each individual will be executed a decoding operation to obtain its CNC tool installation list *Toolist* and RGV material transfer path *RI*, as shown in the first line of Algorithm 2. Contrary to the previous description of the individual entering the maturity stage, only the core information is retained during the process of the individual returning from the maturity stage to the juvenile stage, and other contents will be lost. Specifically, these individuals only retain information on their tool mounting ratios. In other words, the sorting of the internal elements of the CNC tool installation list *Toolist* will be disrupted, and the original RGV material transfer path *RI* will be abandoned. The work of this part is shown in lines 2–3 in Algorithm 2. Before the next iteration, these individuals will experience rebirth from juvenile to maturity, and the lost RGV material transfer path information will be regenerated by the method of one atomic work cycle described above. This part is corresponding to lines 4–5 in Algorithm 2. Next, the RGV material transfer path will be repeatedly made up, so that its dimension is equal to $E(D)$–N_{CNC}. The result will be performed by *Toollist* to perform coding operation, to obtain individual I after the execution of the transdifferentiation strategy, such

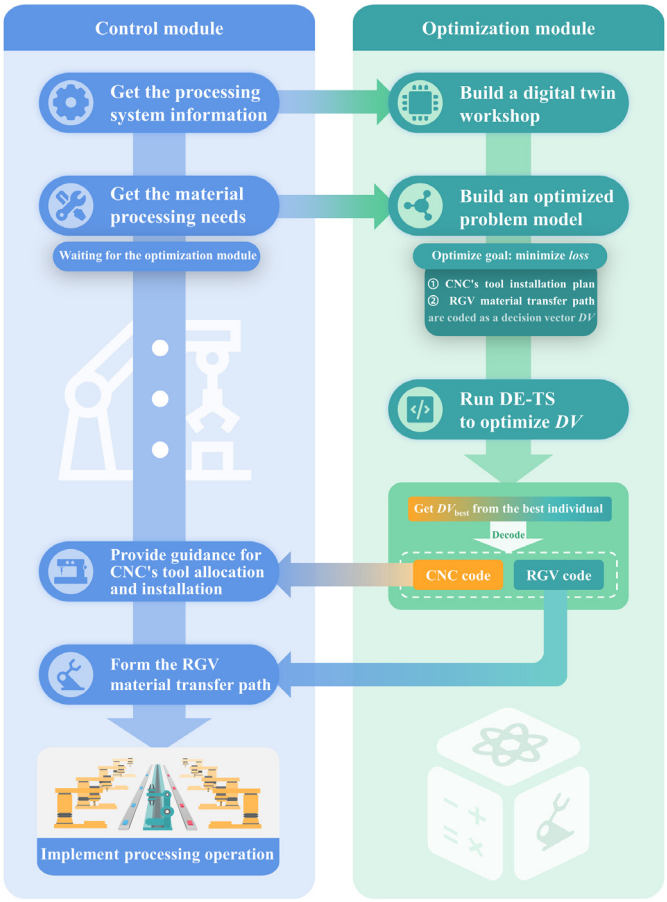

Fig. 6. The relationship between the control module and the optimization module. Among them, the control module and optimization module are identified with blue and green respectively.

as the 6–12 lines of Algorithm 2. Finally, these individuals will be placed in the parent population in the next iteration.

3.5. Analysis of DE-TS

The pseudocode of the DE-TS algorithm proposed in this paper is shown in Algorithm 3. Firstly, the population is randomly initialized, and the number of function evaluation *FE* is set to 0 at this time. This part is corresponding to lines 1–3 in Algorithm 3. Secondly, the population starts iterative optimization. In each iteration, a new round of offspring population is obtained by the standard DE. Afterward, the population is optimized once according to the offspring individual optimization method described in Algorithm 1. This part is corresponding to lines 4–6 in Algorithm 3. Next, the population will be internally sorted in a fit manner and divided into three parts under the influence of environmental pressure α: those with the lowest fitness are abandoned, while those with the highest fitness go directly to the next iteration, and the remaining individuals are saved to the parent population in the next iteration after executing the transdifferentiation strategy. For any individual, its fitness is defined as the value of $1 - loss$. This part is corresponding to lines 7–8 in Algorithm 3. The population split operation is corresponding to lines 9–18 in Algorithm 3. At this time, each individual will perform the corresponding operation according to the order in its population. The 19th line of Algorithm 3 describes the process of assignment of the parent population in the next iteration. The WHILE loop as shown in lines 4–20 in Algorithm 3 will be repeatedly executed until the end conditions are met. When the number of function evaluations reaches *maxFE*, the algorithm will end the iteration and display the best individual, as shown in lines 20–21 in Algorithm 3.

Fig. 6 shows how the control module and the optimization module affect each other. Among them, the arrows in each module are used to illustrate the execution order, and the arrows between different modules show how to interact. First of all, the optimization module will obtain the processing system information from the control module, including the number of CNCs in the processing system, and the maximum continuous processing time of the system, which is used to build a digital twin workshop. Subsequently, the optimization module will obtain material processing requirements information from the control module, including information such as the time of different process stages of the required processing materials. The optimization module will determine the dimension of decision variables and their upper and lower limitation based on the above information, and complete the initialization of the population on this basis. At this time, the optimized module uses the proposed algorithm for optimization operations to return the best individual in the final population as the optimization result. The optimization results obtained will be automatically decoded into CNC code and RGV code. The control module will first obtain the CNC code from the optimization module, which will be used to guide multiple CNCs in the workshop to complete the allocation and installation of the tools. Each CNC will be determined to be responsible for a process and install the processing tool required for it. After the installation is completed, the control module will obtain RGV code from the optimization module, which is used to form a material transfer path for RGV. After the above operations, the machining system is turned on, and the RGV can access all the CNCs in the processing system according to the obtained RGV material transfer path, and automatically place the material to be processed on the processing table of the accessible CNC. For any CNC, whenever a material to be processed is placed on its processing table, the CNC will automatically start processing

Table 1
Parameter settings of all algorithms.

Algorithm	Parameter
IGAR (Fan et al., 2019)	$CR_{\min} = 0.75$, $CR_{\max} = 0.95$, $MR_{\min} = 0.001$, $MR_{\max} = 0.100$
GEO (Mohammadi-Balani et al., 2021)	$AP_{\min} = 0.5$, $AP_{\max} = 2.0$, $CP_{\min} = 1.0$, $CP_{\max} = 0.5$
IMODE (Sallam et al., 2020)	$N_{\min} = 4$, $Rate_a = 2.6$
SHADE (Tanabe and Fukunaga, 2013)	$M_{CR} = 0.5$, $M_F = 0.5$
DE-TS	$\mu = 0.8$, $\alpha = 0.5$
DE (Storn and Price, 1997)	$CR = 0.9$, $F = 0.5$

operations. RGV and CNC will be executed according to the above operations until the maximum processing time is allowed.

Fig. 7 shows the detailed flow chart of the proposed algorithm, in which Fig. 7.a, .b, and .c describe the detailed process of the three modules of population initialization, generation of offspring, and population update, respectively. The DE-TS algorithm will run these three modules in sequence. When the 7.c module is completed, if the maximum number of function evaluations is not reached, return to the 7.b module to continue running, otherwise, end and return the running result.

4. Simulation results and analysis

In this section, all simulation experiments are carried out on a device equipped with an Intel i7-7600U@2.80 GHz dual-core processor. On the one hand, the hardware environment in which the algorithm runs also includes a running memory of 16G and a graphics card with Intel HD Graphics 620. On the other hand, the software platform on which the algorithm runs in Matlab 2020b. The parameter settings of all algorithms are listed in Table 1. Among them, the IGAR algorithm (Fan et al., 2019) proposed by H. Fan et al. is an improved algorithm based on the standard GA (Holland, 1992), and the Golden Eagle Optimization Algorithm (GEO) (Mohammadi-Balani et al., 2021) is an evolutionary algorithm that simulates the multi-stage hunting of golden eagles. Improved multi-operator differential evolution algorithm (IMODE) (Sallam et al., 2020), success-history based parameter adaptation for differential evolution (SHADE) (Tanabe and Fukunaga, 2013) and the DE-TS algorithm proposed in this paper are all improved algorithms based on the standard DE algorithm. At present, the source code of the DE-TS algorithm proposed is available on our project homepage https://github.com/MLNST-JUST/DE-TS, and the three comparison algorithms, such as IMODE, SHADE, and DE, have been integrated into the PlatEMO platform, you can refer to the content of our guidance manual provided in our project homepage to install them. You can also refer to the content of chapter 5 in the guidance manual to run the experiments in this paper again. In addition, the source code of the GEO algorithm has been uploaded to https://www.mathworks.com/matlabcentral/profile/authors/14675656 by its authors.

Among the six algorithms listed in Table 1, the first five algorithms are the main comparison algorithms of this paper. The last algorithm in Table 1 has only been used in the transdifferentiation strategy effectiveness experiment in the 4.1 section to verify whether the transdifferentiation strategy proposed in the algorithm has a positive impact on the effect. For the two algorithms of IGAR and GEO, all their parameters are generated randomly. Their values will be randomly selected between the maximum values and minimum values given by the parameter list. For example, $AP_{\min} = 0.5$, $AP_{\max} = 2.0$ is given in the parameter list of the GEO, indicating that its parameter AP will be randomly determined between [0.5, 2.0]. In addition, the parameters of the remaining four algorithms have given specific values in Table 1. The population size of all algorithms was set to 50 during all experiments. In order to ensure the fairness of the experiment, the maximum function evaluation times of all algorithms are set to 500 times.

Similar to Ding et al. (2020), Fan et al. (2019), Li (2019), Wang et al. (2018, 2019) and Xiao (2019), the simulation experiments in this study are carried out on the RGV and CNC Machining Parameter Dataset (RCMPD). All algorithms use the *loss* as the evaluation indicator for improving the fairness of the experiments. During the simulation experiments, the PlatEMO component developed by the Institute of Bioinspired Intelligence and Mining Knowledge of Anhui University (Tian et al., 2017) was used. Except for the algorithm proposed, the parameter combinations used by the comparison algorithm come from their original papers, or the default parameter combinations provided by the PlateMo platform are called. Part of the visualization work in this article is based on the tools on chiplot.online. Therefore, before using the source code provided in this paper, you need to install the PlatEMO component version 3.3 in MATLAB. This research implements the RDSP series of test problems based on Matlab 2020b, and readers can download these files on the homepage of this project.

Five groups of experiments were conducted in this paper. In the 4.1 section, we performed the transdifferentiation strategy effectiveness experiment. The DE-TS is proposed based on the improvement of the DE. We verify the effectiveness of the strategy introduced in the DE by comparing the evolution curve of the DE-TS and DE. In Section 4.2, we conducted comparative experiments on the three test problems of RDSP1-3 with reference to the experiments set in Ding et al. (2020), Fan et al. (2019), Li (2019), Wang et al. (2018, 2019) and Xiao (2019) to verify the superiority of the proposed algorithm in performance. Furthermore, we also verified the performance of the proposed algorithm and the comparison algorithms under different numbers of CNCs through experiments. In the 4.3 section, we conducted further convergence experiments on the proposed algorithm, and analyzed the convergence of the proposed algorithm and the distribution of *loss*. Furthermore, we conducted an extended simulation experiment in the 4.4 section. The purpose of this experiment is to eliminate the potential impact that fixed test problems may have on the experimental results, and clearly show the distribution of the searching ability of different algorithms in the problem space. Finally, we performed an interpretable experiment in Section 4.5 to analyze the transfer probability matrix of the proposed algorithm's Markov process and the potential information of the RGV material transfer path. The relationship between the complexity of logistics needs and the load of CNC further reveals the inherent reasons for the superiority of the algorithm proposed.

4.1. Transdifferentiation strategy effectiveness experiment

In the experiments of this sub-section, the DE-TS algorithm using the transdifferentiation strategy will be compared with the DE algorithm without this strategy to verify the effectiveness of the transdifferentiation strategy designed in this paper. The experiment is simulated for 30 consecutive days in a processing workshop using RGV for logistics management, to better reflect the performance of different algorithms. Among them, Fig. 8.a shows the loss of the DE-TS and DE algorithms under different numbers of CNCs. Both algorithms have repeated experiments on all datasets, and the mean value is used as their experimental results. This greatly helps to eliminate the influence of system operating parameters within different datasets on the RGV logistics load capacity.

It is not difficult to find that the number of CNCs in the processing system has a certain influence on the *loss* of the initial population — the smaller number of CNCs, the higher *loss* of the initial population. This situation is closely related to the load situation of the RGV — when the number of CNCs increases, the average material logistics requests to be processed by the RGV in the same unit time will be more intensive. This will reduce the average *loss* of the initial population to a certain extent. Fig. 8.a uses circles and triangles to represent the DE-TS and DE algorithms, respectively — the fitness of the initial populations for these two algorithms is almost indistinguishable. However, as the number of iterations increases, the performance of the DE algorithm improves significantly less than DE-TS in any case. Although the *loss*

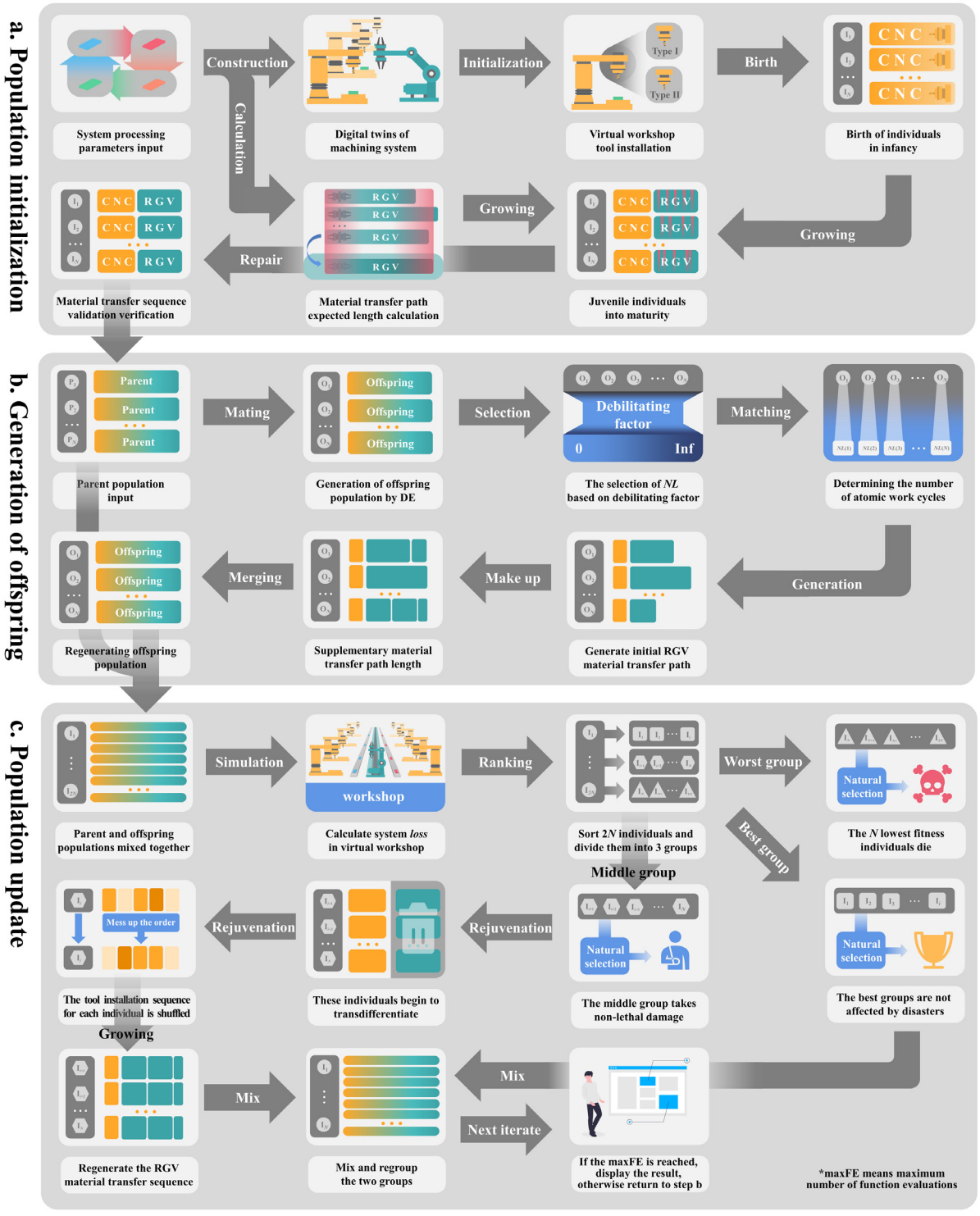

Fig. 7. The graphical flowchart of DE-TS.

of the two algorithms is basically the same in the initial situation, the DE-TS algorithm performs astonishing optimization ability in the iterative process and can achieve fast convergence. Fig. 8.a uses dark blue, green, and yellow for the cases where the number of CNCs is 8, 10, and 12, respectively. In these three scenarios, the *loss* obtained by the DE-TS algorithm is 46.90% lower on average than that of the DE algorithm. This experiment can show that the transdifferentiation strategy can effectively help the algorithm to improve performance. In processing scenarios with different numbers of CNCs installed, its performance is significantly improved compared to DE.

Fig. 8.b shows the *loss* of the above two algorithms in the RDSP series of problems. Both algorithms have repeated experiments in scenarios with different numbers of CNCs, and the average value is used as their experimental results. This helps to eliminate the influence of the complexity of shop floor logistics in different CNC scenarios on the logistics load capacity of the RGV. Compared with Fig. 8.a, the *loss* of

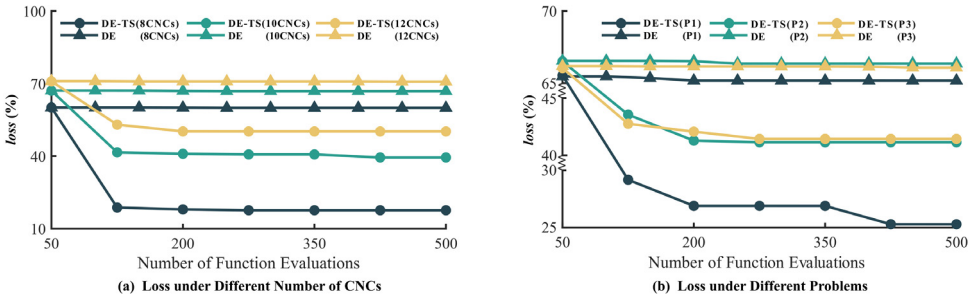

Fig. 8. Results of the effectiveness experiment of the transdifferentiation strategy. Fig. 8.a and .b show the algorithm performance under different numbers of CNCs and different test problems, respectively.

Table 2
The average loss and its standard deviation of the five algorithms in the RDSP series of problems. The optimal algorithm in each scene has been highlighted with a gray background; at the same time, the optimal situation of each algorithm under different CNC numbers is also marked in bold.

Problem	N_{CNC}	D	GEO (Mohammadi-Balani et al., 2021)	IMODE (Sallam et al., 2020)	IGAR (Fan et al., 2019)	SHADE (Tanabe and Fukunaga, 2013)	DE-TS
RDSP1	8	32408	5.9969e-01(2.07e-03)[+]	5.9962e-01(1.30e-03)[+]	**3.1818e-01(9.91e-04)[+]**	5.9921e-01(6.52e-04)[+]	**3.4850e-02(6.76e-03)**
	10	38890	**6.5971e-01(1.06e-03)[+]**	**6.5873e-01(1.44e-03)[+]**	**4.2589e-01(9.83e-04)[+]**	**6.5861e-01(1.05e-03)[+]**	**2.7561e-01(1.81e-02)**
	12	45372	**7.0446e-01(8.28e-04)[+]**	**7.0458e-01(9.79e-04)[+]**	**5.3625e-01(1.95e-03)[+]**	7.0358e-01(5.37e-04)[+]	**4.3704e-01(1.27e-02)**
RDSP2	8	31112	6.2149e-01(1.56e-03)[+]	6.2208e-01(1.84e-03)[+]	3.9093e-01(1.76e-03)[+]	6.2059e-01(1.75e-03)[+]	2.9075e-01(0.00e+00)
	10	37049	6.6541e-01(4.93e-04)[+]	6.6682e-01(9.75e-04)[+]	4.8375e-01(1.57e-03)[+]	6.6600e-01(1.01e-03)[+]	4.3779e-01(1.86e-03)
	12	46298	7.0473e-01(9.26e-04)[+]	7.0472e-01(1.96e-03)[+]	5.8271e-01(4.10e-03)[+]	**7.0319e-01(4.61e-04)[+]**	4.9804e-01(7.10e-03)
RDSP3	8	42731	**5.8198e-01(1.40e-03)[+]**	**5.8198e-01(9.33e-04)[+]**	3.6329e-01(2.25e-03)[+]	**5.8195e-01(2.10e-03)[+]**	2.0818e-01(4.80e-03)
	10	45578	6.8438e-01(2.53e-03)[+]	6.8536e-01(1.49e-03)[+]	5.2644e-01(2.82e-03)[+]	6.8402e-01(1.84e-03)[+]	4.7258e-01(2.15e-03)
	12	56976	7.1622e-01(1.23e-03)[+]	7.1638e-01(2.61e-03)[+]	5.9936e-01(1.57e-03)[+]	7.1642e-01(1.59e-03)[+]	5.4398e-01(1.54e-02)
+/≈/-	-	-	9/0/0	9/0/0	9/0/0	9/0/0	-

all the initial points in Fig. 8.b are very close, which just shows that the number of CNCs has a certain influence on the formation of the initial population, and the fitness of different processing parameters on the initial population impact is minimal. The simulation results show that the DE-TS algorithm still performs significantly better than DE after eliminating the effect of the number of CNCs on the results. The dark blue, green, and yellow in Fig. 8.b are used to represent the performance of the algorithms under the three problems of RDSP1-3, respectively. Under different problems, the *loss* obtained by the DE-TS algorithm is 45.80% lower than the DE algorithm on average — this value is similar to the average 46.90% obtained in the previous experiment, indicating that this may be the average level of the difference between the optimization capabilities of DE-TS and DE. In Fig. 8.b, DE-TS achieves materially better results on RDSP1 than on RDSP2 and RDSP3, and this phenomenon also exists in other algorithms. Although the processing parameters have little effect on the initial population level produced by any algorithm, the difficulty of algorithm optimization varies significantly under different problems—a feature that is independent of which algorithm is used.

4.2. Comparative experiment

In the experiments at this stage, the proposed algorithm will compare the performance with a variety of other algorithms. On an arbitrary test problem, each comparison algorithm was simulated in the RGV workshop for 30 consecutive days and repeated 15 times. The results of this comparative experiment have been summarized in Table 2.

Table 2 contains RDSP1-3 test problems, each of which contains three different machining scenarios with 8, 10, and 12 CNCs. Among them, N_{CNC} is used to represent the number of CNCs, and D is used to represent the dimension of the current problem. First, a side-by-side comparison of different algorithms is performed. In Table 2, there is almost no difference in the algorithmic effects of the three algorithms, GEO, IMODE, and SHADE. The IGAR algorithm is a targeted optimization algorithm for RDSP problems. Compared with the previous 3 algorithms, it works best. However, in all test problems, the effect of the DE-TS algorithm is the best, which shows that the transdifferentiation strategy designed in this paper has a significant positive impact on the RDSP series of problems. In addition, looking at the different scenarios of each test problem individually, when the number of CNCs increases, the complexity of the logistics control in the processing workshop and the dimension of the problem also increases. Likewise, when the number of CNCs increases, the performance of any algorithm shows a decreasing trend. Comparing longitudinally within each algorithm, it is evident that the three algorithms GEO, IMODE, and SHADE all have similar performance on RDSP1 and RDSP3, and all have the worst effect on the RDSP. However, regardless of the number of N_{CNC}, both IGAR and DE-TS algorithms have the best performance on RDSP1, followed by RDSP3 and RDSP2, which is consistent with the conclusion drawn in the previous experiments: the optimization difficulty of the RDSP1 problem is significantly lower than that of the other two problems, and the optimization difficulty of the RDSP2 is slightly stronger than that of the RDSP3. Further, based on the data in Table 2, we calculated the mean values of the output rate F of the four comparison algorithms on 9 test cases, and subtract them from the output rate of DE-TS to obtain the difference between the output rate of DE-TS and the comparison algorithms on each test case. Then, the mean value of these differences was calculated. The result shows that the output rate of DE-TS is 25.68% higher than that of the comparison algorithms on average.

Fig. 9 shows the average performance and running time of all the comparison algorithms included in Table 2 on different test problems. On the one hand, on all test problems, the performances of SHADE, GEO, and IMODE algorithms are very close, which is consistent with the previous results. Among the three algorithms, GEO takes the shortest time, and IMODE takes the second place. On the other hand, the performance of the IGAR and DE-TS algorithms is obviously better than the other three algorithms. In most cases, IGAR runs longer than DE-TS. Except for DE-TS, there is almost no difference in the average *loss* obtained by any algorithm in the same scene. In Fig. 9, the ordinates of the points of the same color and shape are basically the same—this indicates that DE-TS has a stronger ability to explore the solution space. Furthermore, most of the results of all algorithms exhibit a tendency to move to the upper right of the image area when the number of CNCs increases. Many studies have confirmed that evolutionary algorithm has the ability to discover effective rules and strategies, not just random enumeration of the huge search space (Siddique et al., 2021), but how to balance the convergence speed, algorithm effect, and other aspects is

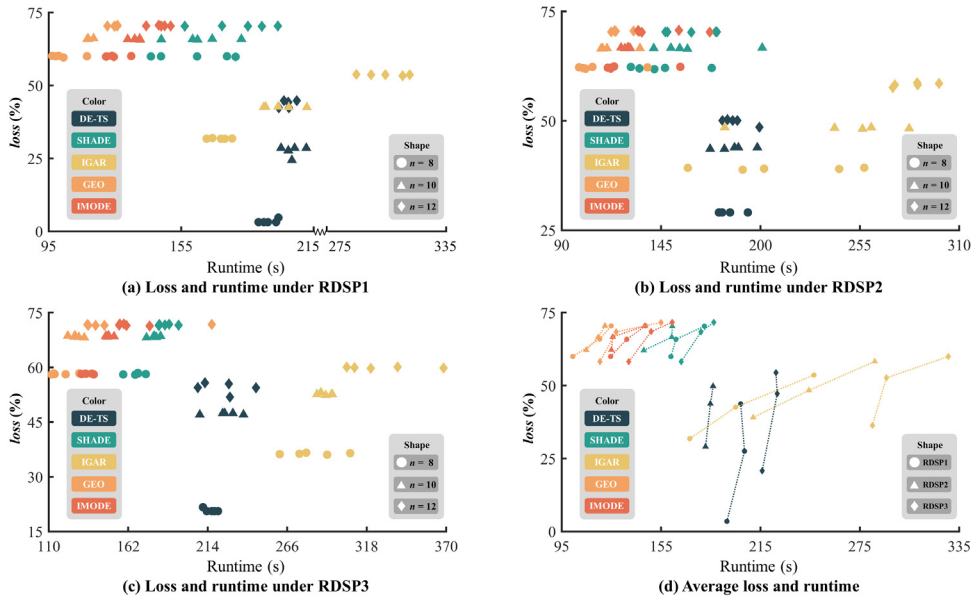

Fig. 9. The relationship between the *loss* and the running time of the five algorithms on the RDSP1-3 test problem. Among them, dark blue, green, yellow, orange, and red are used to represent the five algorithms of DE-TS, SHADE, IGAR, GEO, and IMODE, respectively. In Fig. 9.a–.c, circles, triangles, and diamonds are used to represent scenarios with 8, 10, and 12 CNCs, respectively. In Fig. 9.d, circles, triangles, and diamonds are used to represent the three test questions of RDSP1-3, respectively.

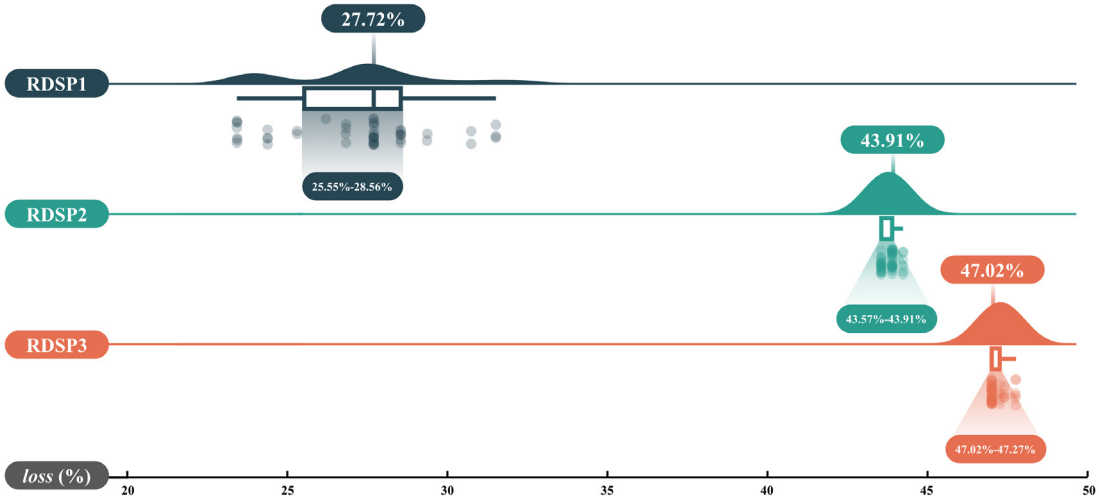

Fig. 10. Convergence test result. The figure shows the distribution of *loss* of DE-TS on RDSP1, RDSP2, and RDSP3 in dark blue, green, and orange respectively.

still a disturbing obstacle (Tang et al., 2015). Fig. 9.d can be obtained by connecting the mean points of any algorithm under different CNC number scenarios. It is simple to discover that the convergence speed of the three algorithms, GEO, IMODE, and SHADE, is slightly better than the other two, but their algorithm performance is always significantly lower than the others. At the same time, although the algorithm performance of IGAR and DE-TS is similar in some cases, the slope of DE-TS in Fig. 9.d is significantly higher than that of IGAR. This phenomenon suggests that the running time of the DE-TS algorithm is less affected by changes in the problem size.

4.3. Convergence experiment

In this section, a high-intensity convergence experiment was performed to analyze the convergence of the proposed algorithm. In this experiment, the DE-TS controls CNC and RGV for collaborative operations in a 30-day processing scenario. At this time, the parameter settings of the DE-TS algorithm are still consistent with Table 1. We use the proposed algorithm to solve the RDSP1-3 when the number of CNCs is 10. Each problem has been run 50 times repeatedly using the DE-TS to understand the distribution of the solution obtained. The result of the convergence experiment has been shown in Fig. 10.

Fig. 10 shows the distribution of the results of 50 repeated experiments of DE-TS on RDSP1-3. The x-axis of Fig. 10 is *loss*, and the 50 times solution results of each problem are drawn in the corresponding area with translucent points. Then, a boxplot is drawn above the point, with the median being marked above the axis, and the interval between the upper quartile and the lower quartile being marked below the axis. We also drew a distribution map of the results above the boxplot to clearly show the distribution of those results. First, we compare the results of this experiment with the previous comparison experiment. According to the results in Fig. 10, we can get that the average *loss* of DE-TS on RDSP1-3 is 27.72%, 43.91%, and 7.02% respectively, which is highly close to the result in Table 2, and the difference between the mean values of *loss* in the two experiments is not more than ± 0.60% of the results in Table 2. This also verifies the convergence of the proposed algorithm to a certain extent. Secondly, according to the results in Fig. 10, we can get that the intervals from the lower

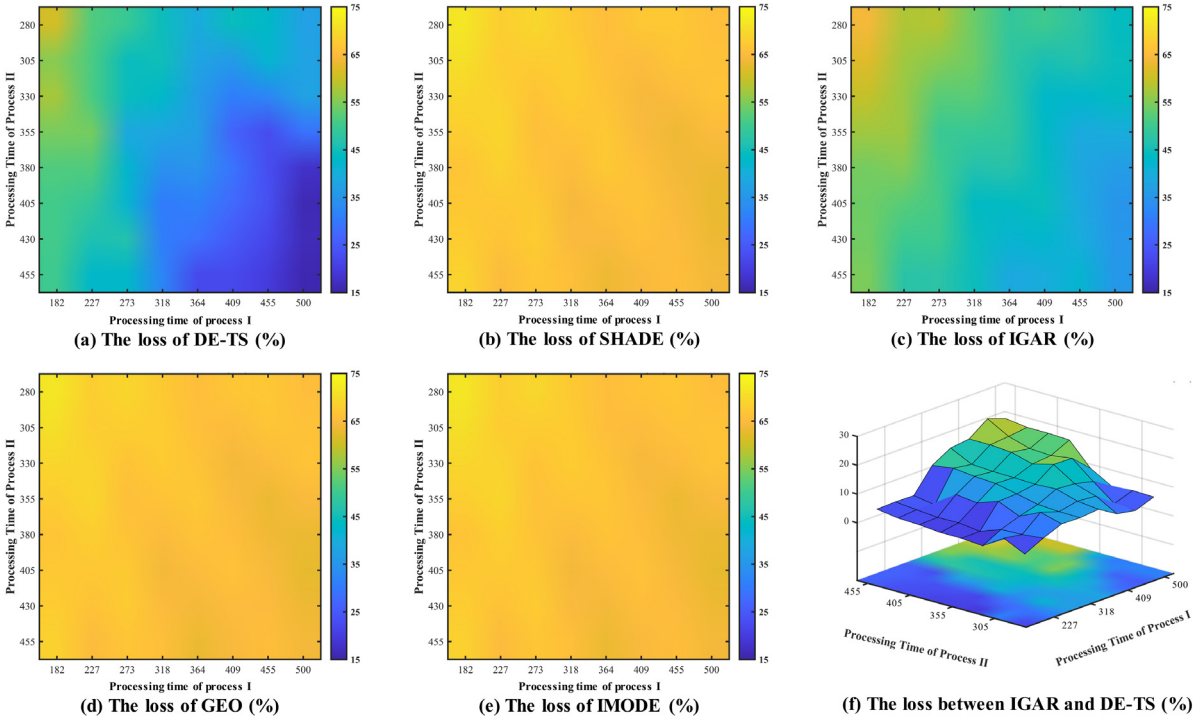

Fig. 11. Algorithm *loss* distribution diagram. Fig. 11.a–.e show the *loss* distributions of the five algorithms on the extended problem space. At the same time, Fig. 11.f also shows the distribution of the difference of the *loss* of the two algorithms with the best performance.

quartile to the upper quartile of the DE-TS on the RDSP1-3 are [25.55%, 28.56%], [43.57%, 43.91%] and [47.02%, 47.27%] respectively. Their interval lengths are 3.01%, 0.34%, and 0.25% respectively. In general, the distribution of the results of the DE-TS on any test problem is very close, which shows that the proposed algorithm has a strong convergence-especially on RDSP2 and RDSP3.

In the boxplot in Fig. 10, the distribution of points does not appear to be continuous, because the calculation function of *loss* is not a continuous function. We have explained the calculation method of *loss* in (3). Each control scheme generated by an algorithm can obtain the corresponding *loss* according to (3), but their distribution is not continuous. In addition, in Fig. 10, the median of RDSP2 is the same as the upper quartile, and the median of RDSP3 is the same as the lower quartile. This also shows that the result of the DE-TS algorithm is highly convergent.

4.4. Extended simulation experiment

In order to explore the limit capability of the proposed algorithm, in the experiment in this sub-section, all algorithms are subjected to extended simulation experiment on an extended set of standard experiment. First, the upper and lower limits of the time it takes for a CNC to complete a class I machining and a class II machining are extracted from all test problems — these data are used to form the experimental scope of the extended simulation experiment. Subsequently, the repeated experiment was performed on all test problems, and the number of CNCs in each experiment was averaged over all possible values. Finally, the mean value of all test results at each data point is used as the value of the corresponding data point to obtain Fig. 11. The purpose of this part of the experiment is to eliminate the potential impact of fixed test problems on the experimental results, and to clearly show the distribution of the solving ability levels of different algorithms in the problem space.

Fig. 11.a–.e show the algorithm *loss* distribution of the five algorithms DE-TS, SHADE, IGAR, GEO, and IMODE on the extended problem space, which is meritorious to study the distribution of solving ability of different algorithms on it. In those heatmaps, the closer the color is to light yellow, the higher the *loss* of the algorithm and the worse the effect of the algorithm; on the contrary, the closer the color is to dark blue, the lower the *loss* of the algorithm and the better the effect of the algorithm. The same as the previous conclusions, the *loss* distributions of the three algorithms SHADE, GEO, and IMODE are relatively similar and are less affected by the processing parameters. When the processing time of both process I and process II decreases rapidly, the material demand in the processing system increases rapidly. At the same time, the effects of these three algorithms are reduced to a certain extent. This phenomenon is reflected in the heatmaps as a lighter yellow in the upper left corner of the image. Furthermore, these three algorithms all perform significantly worse than the other two on the entire extension problem space, which is consistent with the conclusions drawn in previous experiments.

Figs. 11.a and 8.e show the *loss* distributions of DE-TS and IGAR on the extended problem space, respectively. Obviously, DE-TS outperforms IGAR. Similar to other algorithms, the closer to the upper left of the heat map, the denser the logistics requirements of the workshop, and the higher the *loss* of the DE-TS algorithm; the more to the lower right of the heat map, the more dispersed the logistics needs of the workshop, the better the algorithm works. However, in the process of approaching the lower right corner of those images, the change speed of the *loss* of the algorithm in different directions is inconsistent: the route approaching from the left to the lower right corner of those images, while the route approaching from the upper right corner of those images is gentle. This may be due to the fact that the first processing operation has a greater impact on the material in the processing shop, while the subsequent processing operation has relatively less impact on it. Fig. 11.f shows the difference in *loss* between IGAR and DE-TS. It is crystal clear that the whole image is above the *xoy* plane, indicating that the algorithm effect of DE-TS is always better than that of IGAR. When the logistics demand is more intensive, the gap between those algorithms is not significant. However, when the logistics demand is more sparse, although the effects of the two algorithms are overwhelmingly improved, the *loss* difference between the two is increasing, indicating

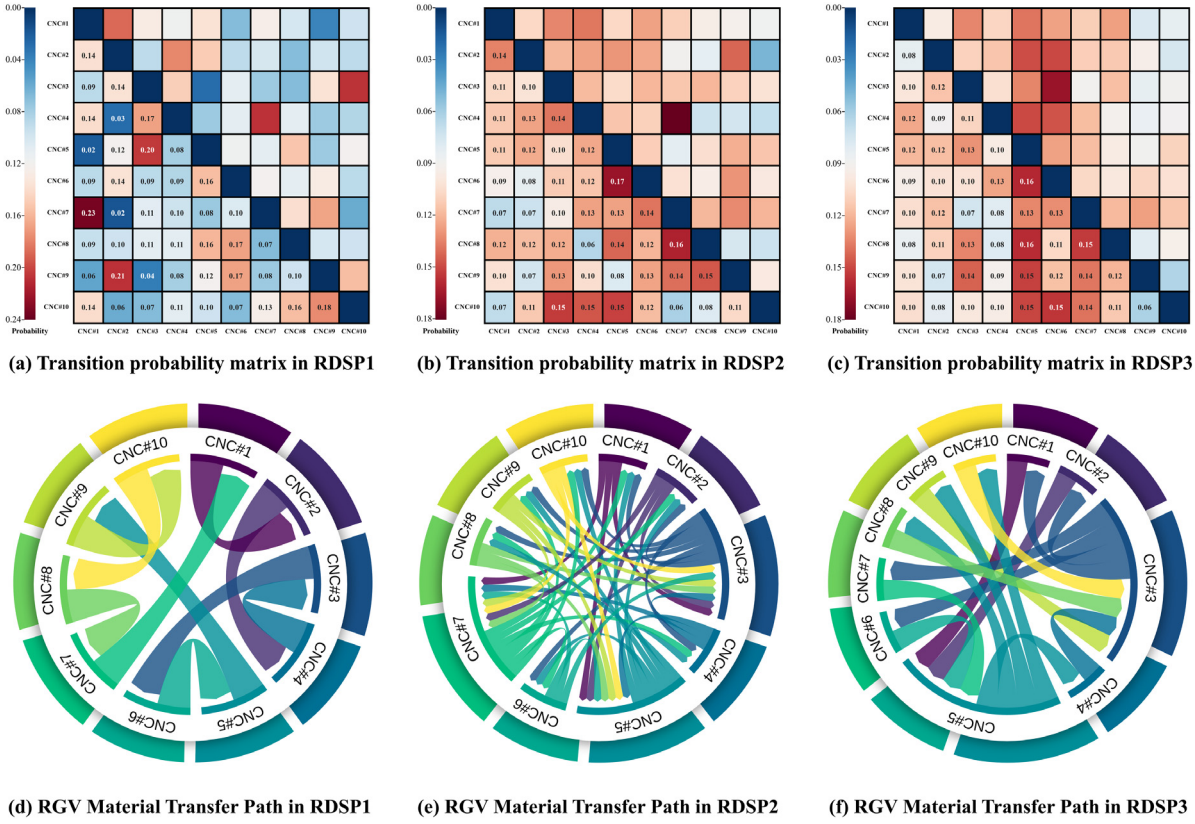

Fig. 12. Material transfer diagram. Fig. 12.a–.c shows the transfer probability matrices of the Markov process on RDSP1-3. At the same time, Fig. 12.d–.f also visually shows the RGV material transfer path information of the proposed algorithm on RDSP1-3.

that DE-TS has better solution space exploration ability and algorithm potential. As has been confirmed in previous experiments, DE-TS is less affected by processing parameters than IGAR.

4.5. explainable experiment

In the previous experiments, the effectiveness of the proposed algorithm has been verified. To explore the internal reasons, we further analyzed the results of DE-TS in this experiment and tried to explain the advantages of DE-TS with these results. In this stage, DE-TS controls CNC and RGV to work cooperatively in the processing scene for 30 consecutive days. At this time, the parameter settings of the DE-TS are still consistent with Table 1. In the scenario where the number of CNCs is 10, we use the proposed algorithm to solve the three test problems of RDSP1-3. Each problem was repeatedly run 50 times with the DE-TS to avoid the impact of contingency on the results. For each test problem, we selected the result with the lowest *loss* in 50 runs, and derived the final population after consuming 500 evaluation times, to further analyze the internal reasons for the superiority of DE-TS. The choice of each step of the RGV material transfer path is treated as a Markov process. By statistical calculation of 50 individuals in the final population, we have obtained the corresponding transition probability matrix of the corresponding Markov process.

The transfer probability matrix of the DE-TS under the three problems of RDSP1-3 is displayed in Fig. 12.a–c. It can be seen that the elements on the diagonal line of these three matrices are 0, which shows that after the RGV visits a certain CNC, it is impossible to immediately access this CNC again. This situation is in line with reality. In fact, after RGV accesses a CNC, it will access a CNC that can handle the next process. If the CNC visited is already the CNC handling the last process, RGV will then access a CNC handling the first process. Therefore, after accessing a CNC, not only this CNC cannot be accessed again immediately, but all other CNC with the same type cannot be accessed immediately. Because different individuals in the population have different tool allocations for CNC, we can only see elements with a value of 0 on the main diagonal of the transition probability matrix. It should be noted that the RGV transition is directional, so the transition probability matrix is not symmetric according to the main diagonal. The closer the numbers between different CNCs are, the closer their actual distances are in reality. In the three transfer probability matrices, most of the elements with large values are distributed near the main diagonal, which also shows that the proposed algorithm has excellent scheduling control capability. When selecting the continuation CNC node for RGV, the physical characteristics can also be taken into account, to reduce the logistics transfer time and improve the processing efficiency of the system. It should be noted that for some CNCs with too large or too small numbers, some elements may not conform to this rule, because these CNCs are distributed at the two poles of the orbit. When the nearby CNC cannot match the processing demand, RGV can only find other CNC to the inside to match the current processing demand.

Furthermore, the information on the RGV material transfer path of the optimal individual of the proposed algorithm on each of RDSP1-3 is shown in Fig. 12.d–.f, to further analyze the characteristics of the proposed algorithm on logistics control. The interior of these three subgraphs shows the RGV transfer path between different CNCs in different colors. These lines start from one CNC and point to another, indicating that there is a corresponding RGV material transfer path in the optimal individual. We have counted the complete RGV material transfer path of the optimal individual. The thicker the line in those figures, the more times the path is executed. The length of the inner ring in those figures also shows the proportional relationship between the number of processing times performed by different CNC. It can be seen that the logistics demand of RDSP1 is the simplest, followed by RDSP3, and the material demand of RDSP2 is the most complex. In RDSP1, the total processing times of different CNCs are nearly the

same, but there are significant differences in RDSP2-3. This is because, in RDSP2-3, the processing time required between different processes is quite different. Therefore, the number and processing times of different types of CNC are also affected. The ring of the outer circle of the subgraph d-f in Fig. 12 shows the proportional relationship between the actual processing time of each CNC. For a CNC, the larger the angle it occupies in the outer ring, the longer the actual processing time it consumes. If the actual processing time of different CNCs is significantly different, it means that some CNCs are idle for a long time under the current control scheme. It can be seen from those figures that DE-TS has excellent control ability. DE-TS can indirectly control the processing load of CNC by controlling the RGV material transfer path, making the processing load of each CNC as consistent as possible. Regardless of the simple or complex logistics requirements, the proposed algorithm can balance the needs of different processes, and make any CNC run under a processing load as high as possible. This further reveals the inherent reason for the superiority of the proposed algorithm.

5. Conclusion

In order to improve the processing efficiency in the workshop using RGV for logistics management, this paper proposes an improved differential evolution algorithm based on the transdifferentiation strategy, called DE-TS. In nature, organisms may heal themselves and continue to survive even after being harmed. Accordingly, in the proposed algorithm, each individual suffers different degrees of damage according to the fitness value and may therefore lose some data. Subsequently, individuals who were not mortally wounded may complete the required data through self-healing and continue to survive. This strategy dedicates to maintaining population diversity. The experimental results show that, compared with other comparative algorithms, the DE-TS algorithm not only achieves the best algorithm performance in the RDSP series of problems, but also has a stable solving ability distribution level on the extended problem space. It is worth noting that the advantages of the proposed algorithm are more significant when the processing time required for the processing operation is longer.

However, in the process of optimization, this paper converts the output of finished materials into the *loss* of the processing system but does not consider the impact of other potential factors on the processing system. In fact, the more distance the RGV moves and the longer the movement time, the more severe the wear of the RGV track and the increase in the power consumption of the workshop. In another work we are doing, the moving distance and moving time of the RGV are also taken into account in the optimization process of another algorithm, in order to ensure the efficient production of finished materials, while extending the life of the track, and reducing power consumption and carbon emissions.

CRediT authorship contribution statement

Yuan-Hao Jiang: Conceptualization, Methodology, Software, Writing – original draft. **Shang Gao:** Supervision, Project administration, Writing – review & editing. **Yu-Hang Yin:** Data curation, Visualization. **Zi-Fan Xu:** Investigation (equal), Formal analysis. **Shao-Yong Wang:** Validation, Investigation (equal).

Declaration of competing interest

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

Data availability

The authors have uploaded the code to their project page on github.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (Grant No. 62176107) and the Postgraduate Research & Practice Innovation Program of Jiangsu Province, China (KYCX21_3487). The author would like to thank Prof. Kezong Tang of Jingdezhen Ceramic University for his valuable guidance on the experimental design of this paper. Huijun Chen of Shanghai Normal University and Na Zhou of Jingdezhen Ceramic University are thanked by the authors for their contributions to the writing of this paper. Ziwei Chen and Chengjian Meng of Jingdezhen Ceramic University are also thanked by the authors for their contributions to the data collection of this paper.

References

Abdulkarim, S.A., Engelbrecht, A.P., 2021. Time series forecasting with feedforward neural networks trained using particle swarm optimizers for dynamic environments. Neural Computing and Applications 33 (7), 2667–2683.

Applegate, D., Cook, W., 1991. A computational study of the job-shop scheduling problem. ORSA J. Comput. 3, 149–156.

Bhaskara, A., Duong, L., Brooks, J., Li, R., McInerney, R., Skinner, M., Pongracic, H., Loft, S., 2021. Effect of automation transparency in the management of multiple unmanned vehicles. Applied Ergon. 90, 103243.

Blot, A., Petke, J., 2021. Empirical comparison of search heuristics for genetic improvement of software. IEEE Trans. Evol. Comput. 25, 1001–1011.

Cai, Y., Wu, S., Zhou, M., Gao, S., Yu, H., 2021. Early warning of gas concentration in coal mines production based on probability density machine. Sensors 21, 5730.

Chen, S., Wang, W., Xia, B., You, X., Peng, Q., Cao, Z., Ding, W., 2021. CDE-GAN: Cooperative dual evolution-based generative adversarial network. IEEE Trans. Evol. Comput. 25, 986–1000.

Cho, J.-H., Kim, Y.-T., 2014. Design of levitation controller with optimal fuzzy PID controller for magnetic levitation system. J. Korean Inst. Intell. Syst. 24, 279–284.

Ding, C., He, H., Wang, W., Yang, W., Zheng, Y., 2020. Optimal strategy for intelligent rail guided vehicle dynamic scheduling. Comput. Electr. Eng. 87.

Dotoli, M., Fanti, M.P., 2005. A coloured Petri net model for automated storage and retrieval systems serviced by rail-guided vehicles: a control perspective. Int. J. Comput. Integr. Manuf. 18, 122–136.

Eguchi, G., Kodama, R., 1993. Transdifferentiation. Curr. Opin. Cell Biol. 5, 1023–1028.

Fan, H., Yu, X., Shu, D.-q., Guan, W.-h., Guo, Z.-h., Feng, S., 2019. An improved genetic algorithm for solving the RGV shop scheduling problem. In: Journal of Physics: Conference Series. IOP Publishing, 012133.

Gao, S., Yu, H., Qiu, L., Cao, C., 2014. The wading across stream algorithm. Int. J. Comput. Appl. 36, 127–132.

Holland, J.H., 1992. Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. MIT Press.

Jain, A.S., Meeran, S., 1999. Deterministic job-shop scheduling: Past, present and future. European J. Oper. Res. 113, 390–434.

Kim, D.H., Lee, I.S., Cha, C.N., 2018. Determination of the pallet quantity using simulation in the FMS for aircraft parts. J. Soc. Korea Ind. Syst. Eng. 41, 59–69.

Kou, X., Xu, G., Yi, C., 2018. Belt-conveyor based efficient parallel storage system design and travel time model analysis. Int. J. Prod. Res. 56, 7142–7159.

Lee, S., De Souza, R., Ong, E., 1996. Simulation modelling of a narrow aisle automated storage and retrieval system (AS/RS) serviced by rail-guided vehicles. Comput. Ind. 30, 241–253.

Lee, D.K., Song, S., Lee, ChanHyeok, Noh, S.D., Yun, S., Lee, H., 2021. Development and application of digital twin for the design verification and operation management of automated material handling systems. Korean J. Comput. Des. Eng. 26, 313–323.

Li, Y., 2019. Scheduling analysis of intelligent machining system based on combined weights. IOP Conf. Ser.: Mater. Sci. Eng. 493.

Loubière, P., Jourdan, A., Siarry, P., Chelouah, R., 2018. A sensitivity analysis method aimed at enhancing the metaheuristics for continuous optimization. Artif. Intell. Rev. 50, 625–647.

Madiouni, R., Bouallègue, S., Haggege, J., Siarry, P., 2019. Epsilon-multiobjective particle swarm optimization-based tuning of sensitivity functions for polynomial control design. Trans. Inst. Meas. Control 41, 3688–3704.

Martina, C., Alessandro, P., Fabio, S., 2018. Modelling of rail guided vehicles serving an automated parts-to-picker system. IFAC-PapersOnLine 51, 1476–1481.

Mohamadi, H.E., Kara, N., Lagha, M., 2022. Heuristic-driven strategy for boosting aerial photography with multi-UAV-aided Internet-of-Things platforms. Eng. Appl. Artif. Intell. 112, 104854.

Mohammadi-Balani, A., Nayeri, M.D., Azar, A., Taghizadeh-Yazdi, M., 2021. Golden eagle optimizer: A nature-inspired metaheuristic algorithm. Comput. Ind. Eng. 152, 107050.

Muhammad, K., Gao, S., Qaisar, S., Abdul, M., Muhammad, A., Usman, A., Aleena, A., Shahid, A., 2018. Comparative analysis of meta-heuristic algorithms for solving optimization problems. In: 2018 8th International Conference on Management, Education and Information (MEICI 2018). Atlantis Press, pp. 612–618.

Ogunsina, K., DeLaurentis, D., 2022. Enabling integration and interaction for decentralized artificial intelligence in airline disruption management. Eng. Appl. Artif. Intell. 109, 104600.

Pezzella, F., Morganti, G., Ciaschetti, G., 2008. A genetic algorithm for the flexible job-shop scheduling problem. Comput. Oper. Res. 35, 3202–3212.

Piraino, S., Boero, F., Aeschbach, B., Schmid, V., 1996. Reversing the life cycle: medusae transforming into polyps and cell transdifferentiation in Turritopsis nutricula (Cnidaria, Hydrozoa). Biol. Bull. 190, 302–312.

Pozna, C., Precup, R.-E., Horváth, E., Petriu, E.M., 2022. Hybrid particle filter–particle swarm optimization algorithm and application to fuzzy controlled servo systems. IEEE Trans. Fuzzy Syst. 30, 4286–4297.

Precup, R.-E., David, R.-C., Roman, R.-C., Petriu, E.M., Szedlak-Stinean, A.-I., 2021. Slime mould algorithm-based tuning of cost-effective fuzzy controllers for servo systems. Int. J. Comput. Intell. Syst. 14, 1042–1052.

Sallam, K.M., Elsayed, S.M., Chakrabortty, R.K., Ryan, M.J., 2020. Improved multi-operator differential evolution algorithm for solving unconstrained problems. In: 2020 IEEE Congress on Evolutionary Computation. CEC, IEEE, pp. 1–8.

Shang, G., 2008. Solving weapon-target assignment problems by a new ant colony algorithm. In: 2008 International Symposium on Computational Intelligence and Design. IEEE, pp. 221–224.

Shang, G., Xinzi, J., Kezong, T., 2007a. Hybrid algorithm combining ant colony optimization algorithm with genetic algorithm. In: 2007 Chinese Control Conference. IEEE, pp. 701–704.

Shang, G., Zaiyue, Z., Xiaoru, Z., Cungen, C., 2007b. Immune genetic algorithm for weapon-target assignment problem. In: Workshop on Intelligent Information Technology Application (IITA 2007). IEEE, pp. 145–148.

Shao, C., Gao, S., Song, X., Yang, X., Xu, G., 2018. Linear representation of intra-class discriminant features for small-sample face recognition. J. Eng. 2018, 1668–1673.

Siddique, A., Browne, W.N., Grimshaw, G.M., 2021. Frames-of-reference based learning: Overcoming perceptual aliasing in multi-step decision making tasks. IEEE Trans. Evol. Comput..

Storn, R., Price, K., 1997. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. J. Global Optim. 11, 341–359.

Tanabe, R., Fukunaga, A., 2013. Success-history based parameter adaptation for differential evolution. In: 2013 IEEE Congress on Evolutionary Computation. IEEE, pp. 71–78.

Tang, K., Li, Z., Luo, L., Liu, B., 2015. Multi-strategy adaptive particle swarm optimization for numerical optimization. Eng. Appl. Artif. Intell. 37, 9–19.

Tang, K., Liu, S., Yang, P., Yao, X., 2021. Few-shots parallel algorithm portfolio construction via co-evolution. IEEE Trans. Evol. Comput. 25, 595–607.

Tian, Y., Cheng, R., Zhang, X., Jin, Y., 2017. PlatEMO: A MATLAB platform for evolutionary multi-objective optimization [educational forum]. IEEE Comput. Intell. Mag. 12, 73–87.

Wang, X., Dong, F., Liu, J., Tan, Y., Hu, S., Zhao, H., 2021. The self-healing of Bacillus subtilis biofilms. Arch. Microbiol. 203, 5635–5645.

Wang, H., Dong, C., Xu, Y., 2018. RGV dynamic scheduling optimization model based on greedy algorithm. Adv. Comput. Signals Syst. 2, 8–10.

Wang, L., Mu, Y., Gao, H., Men, R., 2019. The research on intelligent RGV dynamic scheduling based on hybrid genetic algorithm. J. Phys. Conf. Ser. 1311.

Wang, L., Peng, Z.-p., 2020. Solving energy-efficient distributed job shop scheduling via multi-objective evolutionary algorithm with decomposition. Swarm Evol. Comput. 58, 100745.

Wang, C.-N., Yang, G.K., Hsiung, W.-M., 2014. Application of TRIZ to improve the material handling and productivity in a screw-packaging factory. J. Test. Eval. 42, 1450–1457.

Xiao, H., 2019. RGV dynamic scheduling model based on kruskal algorithm. IOP Conf. Ser.: Mater. Sci. Eng. 612.

Yin, Y.-H., Shen, L.-C., Jiang, Y., Gao, S., Song, J., Yu, D.-J., 2022. Improving the prediction of DNA-protein binding by integrating multi-scale dense convolutional network with fault-tolerant coding. Anal. Biochem. 656, 114878.

Zamfirache, I.A., Precup, R.-E., Roman, R.-C., Petriu, E.M., 2022. Reinforcement learning-based control using Q-learning and gravitational search algorithm with experimental validation on a nonlinear servo system. Inform. Sci. 583, 99–120.

Zhang, G., Ma, X., Wang, L., Xing, K., 2022. Elite archive-assisted adaptive memetic algorithm for a realistic hybrid differentiation flowshop scheduling problem. IEEE Trans. Evol. Comput. 26, 100–114.

Zhang, F., Mei, Y., Nguyen, S., Zhang, M., 2021a. Correlation coefficient-based recombinative guidance for genetic programming hyperheuristics in dynamic flexible job shop scheduling. IEEE Trans. Evol. Comput. 25, 552–566.

Zhang, F., Mei, Y., Nguyen, S., Zhang, M., Tan, K.C., 2021b. Surrogate-assisted evolutionary multitask genetic programming for dynamic flexible job shop scheduling. IEEE Trans. Evol. Comput. 25, 651–665.

Zhang, X., Wang, Z., 2019. Dynamic scheduling model of intelligent rail-guided vehicles based on dynamic programming. In: 2019 International Conference on Intelligent Computing, Automation and Systems. ICICAS, pp. 11–15.