Decision Making: Applications in Management and Engineering
Vol. 6, Issue 2, 2023, pp. 16-42.
ISSN: 2560-6018
eISSN: 2620-0104
DOI: https://doi.org/10.31181/dmame060123042023m

OPTIMIZING PRODUCTION SCHEDULING WITH THE RAT SWARM SEARCH ALGORITHM: A NOVEL APPROACH TO THE FLOW SHOP PROBLEM FOR ENHANCED DECISION MAKING

Toufik Mzili[1]*, Ilyass Mzili[2] and Mohammed Essaid Riffi [1]

[1] Department of Computer Science, Faculty of Science, Chouaib Doukkali University, EI Jadida, Morocco
[2] Department of Management, Faculty of Economics and Management, Hassan First University, Settat, Morocco

Received: 19 January 2023;
Accepted: 15 April 2023;
Available online: 27 April 2023.

Original scientific paper

Abstract: *The Rat Swarm Optimizer (RSO) algorithm is examined in this paper as a potential remedy for the flow shop issue in manufacturing systems. The flow shop problem involves allocating jobs to different machines or workstations in a certain order to reduce execution time or resource use. The objective function is used by the RSO method to optimize the results after mapping the rat locations to task-processing sequences. The RSO method successfully locates high-quality solutions to the flow shop problem when compared to other metaheuristic algorithms on diverse test situations. This research helps to improve the flexibility, lead times, quality, and efficiency of the production system. The paper introduces the RSO algorithm, creates a mapping strategy, redefines mathematical operators, suggests a method to enhance the quality of solutions, shows how successful the algorithm is through simulations and comparisons, and then uses statistical analysis to confirm the algorithm's performance.*

Keywords: *Artificial Rat Swarm Optimization, flow shop problem, scheduling, manufacturing systems, machine processing, job sequence, optimization, metaheuristic algorithms, solution quality, computational efficiency.*

1. Introduction

The manufacturing systems (Zheng et al., 2022) are complex systems (Wang & Magron, 2022) that involve the production and creation of materials with machines, tools, and labor. Ensuring the efficient operation of these systems is crucial to the success and profitability of a business. One of the main challenges facing

* Corresponding author.
E-mail addresses: mzili.t@ucd.ac.ma (T. Mzili), dr.mzili.ilyass@gmail.com (I. Mzili), saidriffi2@gmail.com (M.E. Riffi)

manufacturing systems is the scheduling of tasks on machines, also known as the flow shop problem (Reza & Saghafian, 2005).

This problem involves finding the optimal sequence of operations to process a set of tasks on a set of machines. It arises in manufacturing systems where multiple machines or workstations are used to process a set of tasks, and the tasks must be processed in a specific order and cannot be processed simultaneously on different machines. The objective of the flow shop problem is to find the optimal sequence of operations to process the tasks to minimize the total execution time or to use resources efficiently.

Solving the flow shop problem is a complex optimization (Wang & Magron, 2022) task that requires consideration of multiple variables and constraints. Traditional optimization algorithms may not be sufficient to solve this problem, especially when dealing with large-scale, real-time systems. To address this challenge, researchers have turned to swarm intelligence optimization algorithms.

Swarm intelligence optimization algorithms(Ab Wahab et al., 2015) are a class of optimization algorithms inspired by the self-organizing and decentralized behavior of natural systems, such as flocks of birds (Alaliyat et al., 2014), ant colonies (Blum, 2005), and schools of fish. These algorithms have been widely studied and applied in various fields, including operations research, computer science, and engineering, due to their ability to find good solutions to complex optimization problems in a burnt and efficient manner.

In recent years, swarm intelligence optimization algorithms have received increasing attention as a means of solving the flow shop-scheduling problem, which is a well-known problem in manufacturing systems. The flow shop problem involves scheduling a set of tasks on a set of machines to minimize the completion time of all tasks. Optimizing task scheduling can improve the effectiveness and efficiency of the manufacturing process, thereby reducing costs and increasing competitiveness.

In the continuous flow-scheduling problem, a set of tasks must be processed on a set of machines in a specific order. Each task consists of a sequence of operations, and each operation must be performed on a specific machine. The objective of the scheduling problem is to find a schedule that minimizes the execution time of all tasks. By finding the optimal schedule, manufacturing systems can improve their effectiveness, efficiency, and competitiveness.

There are several variations of the flow shop problem, depending on the specific constraints and objective function. Some common variations include:

– Flow shop with no wait: In this variation (Smutnicki et al., 2022), the machines are assumed to be available for processing at all times, and there is no waiting time between the processing of different jobs.
– Flow shop with total flow time minimization: In this variation (Marichelvam et al., 2017), the objective is to minimize the total processing time of all the jobs.
– Flow shop with makespan minimization: In this variation, the objective is to minimize the time it takes to complete all the jobs, also known as the makespan.
– Flow shop with machine availability constraints: In this variation (Smutnicki et al., 2022), the availability of the machines is taken into account, and the schedule must respect any constraints on the use of the machines.
– Flow shop with job release times: In this variation (Wu et al., 2022), the jobs are released at different times, and the schedule must consider the release times of the jobs.

Solving the flow shop problem requires finding an optimal schedule for the jobs that satisfy the specific constraints and objective function of the problem. This can be a challenging problem due to the complexity of the problem space and the large number of possible schedules that must be evaluated.

There are several reasons why it is important to solve the flow shop problem:

– Improved efficiency: By finding the optimal schedule for the jobs, the flow shop problem can help improve the efficiency of the manufacturing process. This can lead to cost savings and increased profitability.
– Reduced lead times: Scheduling job optimally helps reduce lead times, which is the time it takes for a product to be manufactured and delivered to the customer. Reducing lead times can lead to increased customer satisfaction and competitiveness.
– Improved quality: An optimal schedule can help reduce the risk of errors and defects in the manufacturing process, leading to improved product quality.
– Increased flexibility: Solving the flow shop problem can also help increase the flexibility of the manufacturing system, allowing it to adapt to changing demands and market conditions.

Additionally, the flow shop scheduling problem is of particular importance to businesses because it can help them make informed decisions about their operations. By analyzing and optimizing their production processes, businesses can identify opportunities for improvement and implement strategies to increase efficiency and reduce costs. This can ultimately lead to increased competitiveness and profitability for the business.

In this article, we will review the current state of the art in the application of swarm intelligence optimization algorithms, including the discrete rat swarm optimization algorithm, to solve the flow shop problem. We will discuss the key features of these algorithms and their performance in solving the flow shop problem, as well as the challenges and opportunities for future research in this area. We aim to provide a comprehensive overview of the use of swarm intelligence optimization algorithms, including the discrete rat swarm optimization algorithm, for the flow shop problem and to highlight their potential as a powerful tool for improving the performance of manufacturing systems.

The main contributions of this work concerning production shop scheduling are as follows:

– The introduction of the DRSO algorithm as a solution to the flow shop scheduling problem.
– The development of a mapping strategy to convert real values to discrete values to address the combinatorial nature of flow shop scheduling.
– Redefinition of mathematical operators to solve combinatorial and discrete optimization problems in shop floor scheduling.
– The extension and adaptation of the 2-opt local search heuristic, traditionally used for TSP (Mzili et al., 2022), to solve FSSP.
– Demonstrating the effectiveness of the proposed algorithm through simulations and comparisons with test instances from the OR library.
– The proposal of a statistical analysis using Friedman's test and Holm-Šídák's multiple comparison tests to validate the performance of the proposed algorithm in production shop scheduling.

The organization of this article is as follows: Section 2 presents related work, Section 3 introduces the flow shop problem, Section 4 presents the proposed rat swarm optimization algorithm, Section 5 presents the experimental results, and Section 6 provides a comparison and analysis using the Friedman test, followed by the conclusion and future works.

2. Related works

The flow shop problem is a scheduling problem that involves finding the optimal order of processing a set of tasks on a set of machines to minimize the total processing time. This problem is NP-hard (Tanaev et al., 1994), which means that it is difficult to solve using traditional optimization methods. However, metaheuristics and swarm intelligence algorithms can be used to develop more efficient solutions to the flow shop problem.

Swarm intelligence algorithms are a type of optimization algorithm that is inspired by the self-organizing and decentralized behavior of natural systems, such as flocks of birds, colonies of ants, and schools of fish. These algorithms have been widely studied and applied in various fields, including operations research, computer science, and engineering, due to their ability to find good solutions to complex optimization problems robustly and efficiently.

Popular swarm intelligence metaheuristics that have been used to solve the flow shop problem include ant colony optimization (ACO) (Blum, 2005), particle swarm optimization (PSO)(Zhang et al., 2010), bee colony optimization (BCO) (Huang & Lin, 2011), and the artificial fish swarm algorithm (AFSA) (Babaee et al., 2020). In addition to swarm intelligence algorithms, other types of metaheuristics have also been proposed and applied to solve the flow shop problem.

Iterative improvement-based metaheuristics generate solutions through iterative improvements. The IIGA algorithm (Pan et al., 2008) uses a constructive heuristic and an acceptance criterion to generate and select the best solution for the next iteration. The DPSOVND algorithm (Pan et al., 2008)is designed to minimize both the makespan and total flow time for a shop floor scheduling problem. The TMIIG algorithm (Ding et al., 2015) is a modified version of the iterated greedy algorithm that incorporates a Tabu-based reconstruction strategy and a neighborhood search method involving insertion, permutation, and double-insertion moves to solve the no-wait job shop-scheduling problem with a scope criterion. The NEH (Nawaz, Enscore, and Ham) algorithm (Liang et al., 2022) is a heuristic method for minimizing the execution time in a continuous flow shop with infinite storage at each stage.

Hybrid metaheuristics combine several approaches to leverage individual strengths and overcome their weaknesses. The NEH-NGA algorithm (Liang et al., 2022)combines the NEH heuristic and the niche genetic algorithm to create a hybrid optimization method to solve scheduling problems. The SSO algorithm (Kurdi, 2021)is based on the collaborative behavior of social spider colonies, which involves interactions between males and females performing various tasks. The SCE-OBL algorithm (Kurdi, 2021) combines the SCE algorithm with adversarial learning. The CLS-BFE algorithm (Kurdi, 2021) combines chaotic local search with bacterial foraging principles to search for optimal solutions. The CSO algorithm (Li & Yin, 2013) combines cuckoo search with Levy flights, a random search technique based on the probability distribution of Levy flights observed in nature.

Nature has inspired many metaheuristic algorithms, such as the BAT algorithm (Bellabai et al., 2022), which is inspired by the echolocation system of bats to solve problems. The HMSA algorithm (Marichelvam et al., 2017) combines elements of the Monkey Search algorithm with other techniques to solve the flow shop problem. The DWWO algorithm (Ding et al., 2015) is designed to solve the NWFSP with a focus on minimizing the makespan, and it has five phases. Propagation and breaking operations are based on insertion.

Evolution-inspired metaheuristics use the principles of natural selection and genetics to simulate the evolutionary process. The SGA algorithm (Liang et al., 2022) uses the principles of natural evolution, such as reproduction, mutation, and selection, to search for the optimal solution to a given problem. The GA algorithm (Arik, 2021) is another type of optimization algorithm that draws on the principles of natural evolution and genetics. These algorithms are often used to solve optimization problems, including the flow shop problem.

3. Flow shop problem

Flow shop scheduling is a well-known problem in the field of operations research and manufacturing systems. It can be formalized as an optimization problem whose objective is to minimize the total processing time of a set of tasks on a set of machines. The problem can be formulated as follows:

$$Minimize: \sum_{j=1}^{m}\sum_{i=1}^{n} P_{ij}\, x_{ij} \qquad , \qquad x_{ij} \in \{0,1\}, \forall i,j \tag{1}$$

Subject to:

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad \forall i \tag{2}$$

$$\sum_{j=1}^{m} x_{ij} \leq 1 \qquad \forall i \tag{3}$$

Where:

n is the number of jobs

m is the number of machines

P_{ij} is the processing time for job j on machine i

x_{ij} is a binary decision variable that is 1 if job j is processed on machine i and 0 otherwise

The first constraint ensures that each job is assigned to exactly one machine, and the second constraint ensures that each machine can only process one job at a time. The third constraint indicates that the decision variables are binary.

The objective of the optimization problem is to find the values of the decision variables (x_{ij}) that minimize the total processing time, subject to the constraints. This can be achieved using optimization algorithms, such as linear programming, mixed integer programming, or metaheuristics such as swarm intelligence algorithms.

3.1. Importance of solving the flow shop problem in manufacturing systems

The Flow shop problem is a major challenge in manufacturing systems. It involves planning a sequence of operations for a set of tasks in a specific order through a series of machines. This problem requires effective planning and optimization to minimize production time, reduce costs, and improve productivity.

Solving the FSSP problem can have significant benefits for manufacturing systems, including:

1) Improved efficiency: By optimizing the production schedule, manufacturing systems can operate more efficiently, reducing production time and increasing output.

2) Reduced costs: An optimized production schedule can reduce the need for overtime, excess inventory, and other expenses, resulting in significant savings.

3) Increased competitiveness: Manufacturing systems that can produce goods more efficiently and cost-effectively are more competitive in the marketplace.

4) Improved customer satisfaction: A well-optimized production program can help meet customer demand and ensure on-time delivery, which improves customer satisfaction.

Therefore, solving the flow shop problem is of great importance in manufacturing systems and can have significant benefits for companies.

Figure 1 shows the Gantt chart for 5 tasks and 4 machines.

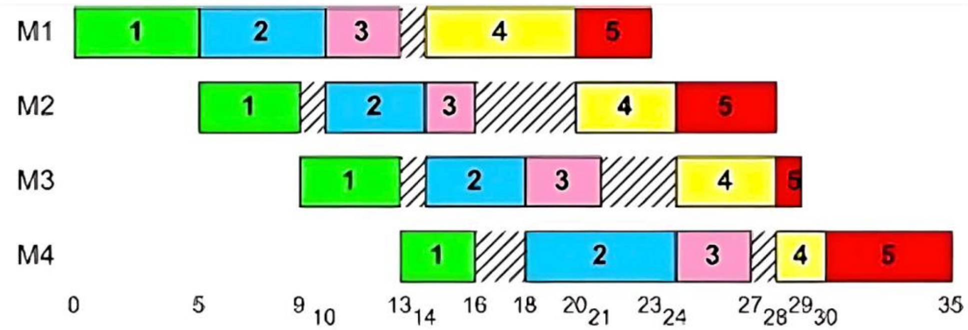

Figure 1. The Gantt Chart Example for 5 jobs and 4 machines

4. Proposed Rat swarm algorithm

Rat Swarm Optimization (RSO) (Mzili et al., 2022) is a metaheuristic algorithm inspired by the behavior of rat swarms and their ability to find food sources efficiently. In particular, the RSO algorithm is inspired by how rat swarms can adapt to changing environments and use their collective intelligence to locate and capture prey.

In the RSO algorithm, a population of "rats" is used to represent potential solutions to the optimization problem. Each rat is associated with a set of decision variables that represent a potential solution to the problem. The rats move through the search space, exploring different solutions and updating their position according to the quality of the solutions found.

4.1. Mathematical modeling of the RSO algorithm

The rat swarm optimization (RSO) algorithm consists of two main phases: exploration and exploitation.

To model the behavior of rats when they search for and capture prey, specific equations are used in the algorithm. These equations allow the rats to locate and capture prey effectively and efficiently while optimizing the position or solution of the prey in the search space.

- **Pursuit of prey (Exploration phase)**

The pursuit behavior of rats that update their position according to the best personal position found by the best searcher in the group. Parameters A and C provide a balance between exploration and exploitation, allowing rats to search for and capture prey efficiently.

This behavior is described by the following equation.

$$P = A * P(t) + C * \left(Pbest(t) - P(t)\right) \quad (4)$$

Where P(t) represents the position of the rat at time t, P(t-1) represents the position of the rat at the previous time step, and Pbest(t) represents the best position of the rat at time t.

$$A = R - \rho\left(\frac{R}{Max_{Iteration}}\right) \ . \ \ 1 \le R \le 5 \quad (5)$$

$$\rho = 1.2.3. \ldots . Max_{Iteration} \quad (6)$$

Therefore, parameters A and R are responsible for balancing exploration and exploitation during the iteration process. They are sensitive to finding a good balance between the two, and their values are randomly generated between 1 and 5 for A and 0 and 2 for R. This helps the rats effectively search for and capture their prey while also optimizing the solution or position of the prey.

- **Fighting prey (exploitation phase)**

The rats attack the target prey detected in the previous phase. However. The prey often tries to escape from dangerous situations or to defend itself against this attack.

In this case. A deadly battle ensues between the rats and the prey and. in some cases. Ends with the death of some rats.

Therefore, the fight between the rats and their prey is mathematically described by the formula below:

$$P(t+1) = |Pbest(t) - P| \quad (7)$$

This equation represents the exploitation phase of the rats, where they accept the position and evaluation of the prey that they have found and fought with. P(t+1) represents the updated position of the rat at the current time step, and Pbest represents the best position or solution found by the rats so far. The absolute value function ensures that the updated position of the rat is always a positive value, regardless of whether Pbest is greater or less than P(t).

$$F(t) = f\left(P(t)\right) \quad (8)$$

Where F(t) represents the evaluation or value of the prey at time t, and f(P(t)) represents the position of the prey at time t.

The value of the prey, represented by F(t), can be determined using a suitable evaluation function, such as the fitness function in an optimization problem. The position of the prey, represented by f(P(t)), can be used to update the personal and global best positions of the rats in the swarm.

4.2. Using the RSO algorithm to solve the flow shop problem

Solving the flow shop-scheduling problem using the RSO algorithm requires the definition of a set of discrete operators that the rats can use to move through the search space. These operators can consist of swapping the position of two tasks in the calendar, inserting a new task into the calendar, or deleting a task from the calendar. The rats then use these operators to explore different scheduling configurations and update their positions based on the quality of the solutions found.

In RSO, a set of "virtual rats" search for an optimal solution by moving through the problem space and adjusting their movement according to the positions of other rats. The rats are guided by a "rat king", who is a virtual leader who guides the movement of the rats toward the optimal solution.

To use RSO to solve the flow shop problem, the following steps can be taken:

1) Define the problem: Clearly define the problem to be solved, including the number of tasks, the number of shops, and any constraints or requirements that need to be addressed.

2) Initialize the population: Create a population of rats that will represent potential solutions to the problem. Each rat will be assigned a set of tasks to perform in a specific order.

3) Evaluate the fitness of each rat: Calculate the fitness of each rat in the population by evaluating the effectiveness of the order of the tasks they have been assigned. The fitness of each rat will be based on measures such as total processing time, number of delays, and overall system efficiency.

4) Selection of the fittest rats: Select the fittest rats from the population using the objective function. These fittest rats will be used to create the next generation of rats.

5) Generate new rats: Generate new rats from the fittest rats using equation (8) by replacing the mathematical operators with other discrete operators such as crossover and mutation. Since this optimizer is designed to solve continuous and linear optimization problems, it cannot be used directly to solve discrete optimization problems. Therefore, several modifications must be made.

For this equation, $P = A * P(t) + C * \left(Pbest(t) - P(t)\right)$, the mathematical operators will be redefined for the flow shop problem.

- $Pbest(t) - P(t)$: The operator of subtraction between two rat positions will be changed in our case to a list of swaps to be performed on a sequence of jobs $P(t)$ to obtain the first sequence list $Pbest(t)$.
- $C * \left(Pbest(t) - P(t)\right)$: This operation between a real [0,1] and a list of swaps will be defined to manipulate and reduce the number of swaps generated by the previous equation.
- $P(t-1) + C * \left(Pbest(t) - P(t)\right)$: The addition operation allows for the final number of possible swaps to be applied to a sequence of jobs.

These changes will be clarified in the example below in the following Figure 2:

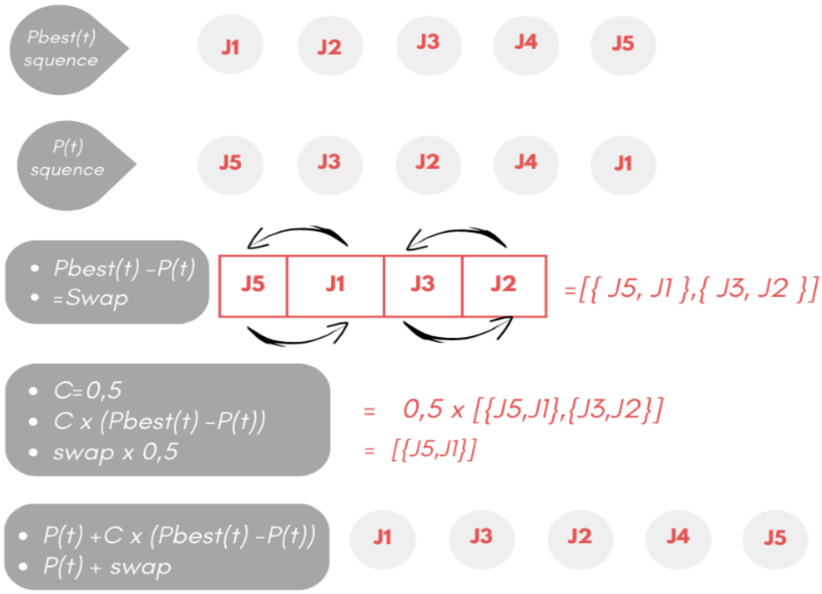

Figure 2. Example of Discrete Operators for Flow Shop Problem

6) The new rats will represent potential new solutions to the virtual workshop problem.

7) Apply the 2-opt local search algorithm to improve each solution: The 2-opt algorithm is primarily used to solve the traveling salesman problem (TSP); however, it can be adapted and extended to address the flow shop (FSSP). The algorithm consists of selecting two non-adjacent edges in the schedule and swapping the order of the tasks between them. After the swap, calculate the new makespan and, if it is greater than the current solution, keep the updated solution. Run this process iteratively several times to gradually refine the quality of the solution.

Algorithm 1 2-opt for Flow Shop Problem

```
procedure 2-OPT(schedule, tasks, machines)
    best_schedule ← schedule
    best_makespan ← calculate_makespan(schedule, tasks, machines)
    for i ∈ 1 . . . |schedule| do
        for j ∈ i + 1 . . . |schedule| do
            new_schedule ← swap(schedule, i, j)
            new_makespan ← calculate_makespan(new_schedule, tasks, machines)
            if new_makespan < best_makespan then
                best_schedule ← new_schedule
                best_makespan ← new_makespan
            end if
        end for
    end for
    return best_schedule
end procedure
```

8) Evaluate the fitness of the new rats: Calculate the fitness of the new rats and add them to the population.

9) Repeat the process: Continue to repeat the process of selecting the fittest rats, generating new rats, and evaluating their fitness until the optimal solution is found or a predetermined number of iterations has been reached.

The following is the description of the final algorithm.

Algorithm 2 Rat Swarm Algorithm for Flowshop Scheduling

```
procedure RatSwarm(J, M, P, S)
    n ← |J|                                                   ▷ Number of jobs
    m ← |M|                                                   ▷ Number of machines
    Initialize RSO parameters: A, C, and R.                   ▷ RSO parameters
    X ← Initialize population of rats                         ▷ Population of rats
    best ← Initialize best solution                           ▷ Best solution found so far
    while (k < T ) do                                         ▷ Iterate for a maximum of T iterations
        for i ← 1 to n do                                     ▷ Iterate over all rats
            x_i ←Update the positions of current search agents Eq. (7)
            x_i ← 2-opt(x_i)                                  ▷
            f_i ← Evaluate fitness of solution x_i            ▷ Fitness of solution x_i
            if f_i < f_best then                              ▷ If new solution is better than current best
                best ← x_i                                    ▷ Update best solution
            end if
            X ← Update population of rats                     ▷ Update population of rats
        end for
    end while
    return best                                               ▷ Return best solution found
end procedure
```

5. Experimental results

The DRSO algorithm has been applied to more than 150 instances of the OR library and the results are presented in Tables (2-7). These tables indicate the instance name ("Instance"), the number of tasks (n) and machines (m) for each instance ("n×m"), the best result proposed by other algorithms ("BKS"), the best results obtained by the different methods ("Best"), and the average results ("Average"). The column "PDav(%)" indicates the percentage deviation of the average solution length from the optimal solution length, calculated using equation 9:

$$PDav(\%) = \frac{(Average - BKS\,) \times 100\%}{BKS} \qquad (9)$$

In the "PDav(%)" column, values of 0.00 are highlighted in bold when all solutions found in the 20 trials are equal to the length of the best-known solution. Values less than 0.00 are highlighted in bold and blue if the average of the solutions found in all trials is less than the length of the best-known solution. Table 1 shows the initial discrete RSO parameters.

Table 1. Parameters of Discrete RSO

PARAMETER	VALUE
THE POPULATION OF RAT SIZE: N	100
A	A random value between [1, 5]
R	A random value between [0, 1]
NB ITERATION	1000

To conduct a comprehensive evaluation of DRSO, it is necessary to compare it with other problem-solving algorithms or methods. A wide range of metaheuristics must be selected to ensure a thorough and detailed analysis of DRSO's strengths and weaknesses compared to other algorithms and to identify the situations in which DRSO performs best. The metaheuristics chosen for comparison include IIGA, DPSOVND, TMIIG, DWWO, BAT, TLBO, SGA, HMSA, NEH, NEH-NGA, SSO, SCE-OBL, CLS-BFE, ACGA, and CSO. This diverse set of metaheuristics will provide a comprehensive basis for comparison and will help to assess the effectiveness of DRSO relative to other optimization methods.

Figure 3 shows the convergence curves of several different algorithms on four instances of Ta001, Ta002, Ta021, and Ta031 in the context of the production-scheduling problem. The curves represent the performance of each algorithm in the four different instances.

The horizontal axis in the Figure represents the number of iterations required to reach the optimal value of the objective function, while the vertical axis represents the value of the objective function.

Examination of the curves shows that the DRSO algorithm converges quickly compared to the other algorithms.

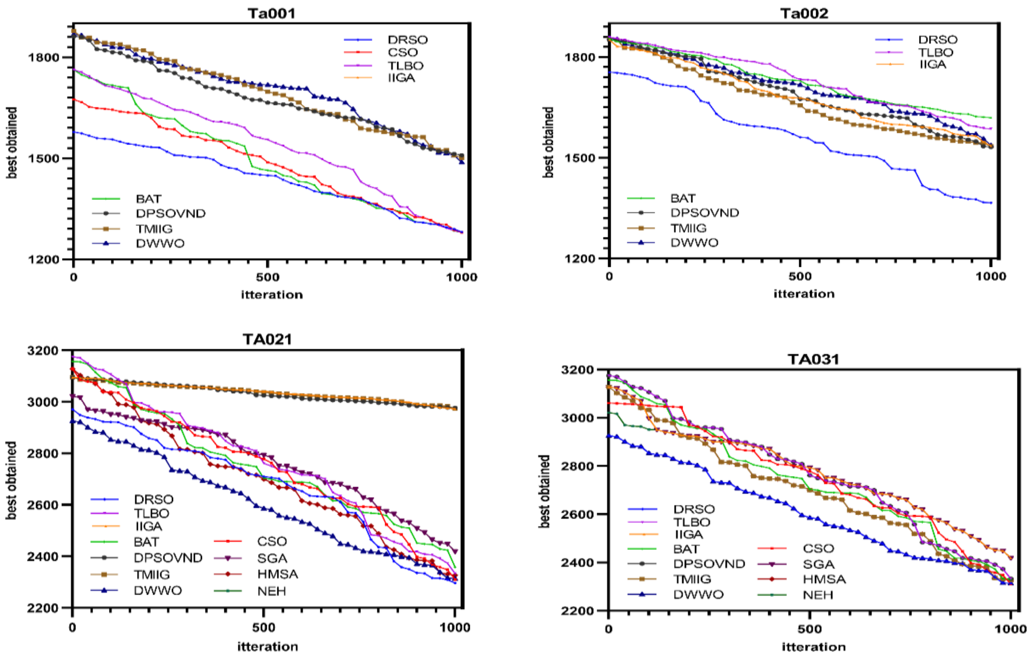

Figure 3. Convergence curve for four instance.

Table 2. Comparison of DRSO, IIGA, DPSOVND, TMIIG, and DWWO Metaheuristics

	DRSO		IIGA		DPSOVND		TMIIG		DWWO	
Instance	Best	Average	Best	Average	Best	Average	Best	Average	Best	Average
Ta001	1278	1278	1486	1486	14 86	1 486	1486	1486	1486	1486
Ta002	1359	1359	1528	1528	1528	1528.6	1528	1528	1528	1528
Ta003	1081	1081	1460	1460	14 60	1 460	1460	1460	1460	1460
Ta004	1293	1293	1588	1588	15 88	1 588	1588	1588	1588	1588
Ta005	1235	1235	1449	1449	14 49	1 449	1449	1449	1449	1449
Ta006	1195	1195	1481	1481	14 81	1 481	1481	1481	1481	1481
Ta007	1239	1239	1483	1483	14 83	1 483	1483	1483	1483	1483
Ta008	1206	1206	1482	1482	1482	1483.2	1482	1482	1482	1482
Ta009	1230	1230	1469	1469	14 69	1 469	1469	1469	1469	1469
Ta010	1108	1108	1377	1377	13 77	1 377	1377	1377	1377	1377
Ta011	1582	1582	2044	2044	20 45	2 045	2044	2044	2044	2044
Ta012	1659	1659	2166	2166	21 66	2 166	2166	2166	2166	2166
Ta013	1496	1496	1940	1940	1940	1940.4	1940	1940.4	1940	1940
Ta014	1377	1377	1811	1811	18 11	1 811	1811	1811	1811	1811
Ta015	1419	1419	1933	1933	19 33	1 933	1933	1933	1933	1933
Ta016	1397	1397	1892	1892	18 92	1 892	1892	1892	1892	1892
Ta017	1484	1484	1963	1963	19 63	1 963	1963	1963	1963	1963
Ta018	1538	1538	2057	2058.6	20 57	2 057	2057	2057	2057	2057
Ta019	1593	1593	1973	1973	19 73	1 973	1973	1973	1973	1973
Ta020	1591	1591	2051	2051	20 51	2 051	2051	2051	2051	2051
Ta021	2297	2298,33	2973	2973	2973	2973.8	2973	2973	2973	2973
Ta022	2100	2100	2852	2852	28 52	2 852	2852	2852	2852	2852
Ta023	2326	2326	3013	3019.4	30 13	3 013	3013	3019.4	3013	3014.3
Ta024	2223	2223	3001	3001	3001	3003.4	3001	3001	3001	3001
Ta025	2291	2291	3003	3003	30 03	3 003	3003	3003	3003	3003
Ta026	2226	2227,20	2998	2998	29 98	2 998	2998	2998	2998	2998
Ta027	2273	2275,27	3052	3052	30 52	3 052	3052	3052	3052	3052
Ta028	2200	2200,83	2839	2839	2839	2849.4	2839	2839	2839	2839
Ta029	2237	2237,26	3009	3009	30 09	3 009	3009	3009	3009	3009
Ta030	2178	2180,24	2979	2979	29 79	2 979	2979	2979	2979	2979
Ta031	2724	2726,75	3209	3222	3169	3178.4	3161	3162.4	3170	3171.8
Ta032	2834	2837,25	3469	3474.4	3444	3450.4	3440	3441	3441	3444.5
Ta033	2621	2621,52	3254	3262.8	3226	3234.4	3213	3216	3218	3231.8
Ta034	2751	2752,84	3366	3374.8	3348	3351.8	3343	3346.6	3349	3350.8
Ta035	2863	2865,97	3398	3406	3370	3374.8	3361	3364.6	3376	3376.7
Ta036	2829	2831,12	3371	3382.2	3354	3362.8	3346	3347.6	3352	3356.2
Ta037	2725	2726,77	3257	3267.6	3244	3250.8	3234	3235.8	3243	3245.3
Ta038	2683	2685,97	3266	3275.2	3247	3257.8	3241	3242.4	3239	3240.3
Ta039	2552	2552	3115	3123.2	30 87	3 092	3075	3078.8	3078	3086.8
Ta040	2782	2782	3373	3377.2	3336	3344.8	3322	3327.2	3330	3336.5
Ta041	2991	2991	4303	4311.2	4284	4298.6	4274	4276.8	4274	4281.5
Ta042	2867	2867	4197	4201	4193	4212.2	4179	4185	4180	4184.5
Ta043	2839	2839	4110	4124	41 19	4 128	4099	4107.6	4099	4105.5

	DRSO		IIGA		DPSOVND		TMIIG		DWWO	
Instance	Best	Average	Best	Average	Best	Average	Best	Average	Best	Average
Ta044	3063	3063	4432	4439.2	4411	4422.8	4399	4405	4407	4405.7
Ta045	2976	2976	4336	4347	4334	4342.8	4324	4330.4	4324	4324.7
Ta046	3006	3006	4312	4330	4311	4316.4	4290	4297.2	4294	4295.2
Ta047	3093	3093	4433	4441.4	4435	4447.6	4420	4429.6	4420	4420
Ta048	3037	3037	4353	4357.6	43 31	4 341	4321	4327	4323	4323.3
Ta049	2897	2897	4190	4194.8	4162	4169.2	4158	4164.2	4155	4161.7
Ta050	3065	3065	4299	4301	42 87	4 290	4286	4286.2	4286	4286.2
Ta051	3875	3875	6144	6154.6	6165	6178.8	6129	6139.6	6129	6138.5
Ta052	3715	3715	5748	5762.2	5730	5751.8	5725	5741	5725	5733.5
Ta053	3668	3668	5879	5907	5881	5898.6	5873	5882.4	5862	5865.5
Ta054	3752	3752	5797	5802.6	5802	5813.4	5789	5791.4	5789	5790.7
Ta055	3635	3635	5924	5930.6	59 08	5 923	5886	5899.4	5886	5893.5
Ta056	3698	3698	5904	5912.6	5886	5901.4	5874	5883.4	5871	5874.3
Ta057	3716	3716	6004	6012	5968	5991.4	5968	5974	5969	5974
Ta058	3709	3709	5947	5970.2	5937	5977.2	5940	5945.4	5926	5930.5
Ta059	3765	3765,602	5881	5900.6	5889	5908.6	5876	5883.2	5876	5876
Ta060	3777	3779,266	5970	5982	5959	5971.6	5959	5959	5958	5958.8
Ta061	5493	5493	6563	6586.4	6458	6464.6	6397	6413.4	6433	6438.3
Ta062	5268	5268,79	6409	6428.2	6268	6292.4	6246	6252.2	6268	6285.5
Ta063	5175	5176,656	6254	6292.8	61 72	6 185	6133	6135.8	6162	6164.2
Ta064	5014	5014,401	6173	6184.8	6071	6085.8	6028	6031.4	6055	6050.7
Ta065	5250	5253,36	6319	6358.4	6244	6261.2	6206	6217.8	6221	6223.3

Table 3. Comparison of DRSO, BAT, and TLBO for FSSP Test Problems: Computational Results

		DRSO			BAT			TLBO		
INSTANCES	BKS	Best	Average	PDav	Best	Average	PDav	Best	Average	PDav
Ta 001	1278	1278	1279,022	0.08	1278	1284.9	0.5399	1278	1287.2	0.7199
Ta 011	1582	1582	1584,057	0.13	1609	1623.3	2.6106	1586	1606	1.5171
Ta 021	2297	2297	2301,364	0.19	2323	2355.4	2.5424	2325	2344.7	2.0766
Ta 031	2724	2724	2726,452	0.09	2724	2725.6	0.0587	2724	2729.4	0.1982
Ta 041	2991	2991	2998,178	0.24	3119	3110.6	3.8449	3120	3141.4	5.0284
Ta 051	3771	3771	3772,886	0.05	4001	4021.9	6.6534	3986	4029.7	6.8602
Ta 061	5493	5493	5507,831	0.27	5493	5496.4	0.0619	5493	5499.4	0.1165
Ta 071	5770	5770	5776,347	0.11	5808	5819.6	0.8596	5887	5928.7	2.7504
Ta 081	6286	6286	6286,629	0.01	6485	6527.2	3.8371	6549	6617.8	5.2784
Ta 091	10868	10868	10887,56	0.18	10942	10942	0.6809	10979	11033	1.5182
Ta 101	11294	11294	11319,98	0.23	11600	11622.5	2.9086	11855	11940	5.7199
Ta 111	26189	26189	26264,95	0.29	26612	26622.6	1.6557	27377	27492	4.9754

Table 4. Comparison of DRSO, SGA, and HMSA on FSSP Test Problems: Computational Results

Instance	(nxm)	BKS	DRSO		SGA		HMSA	
			Best	PDav (%)	Best	PDav (%)	Best	PDav (%)
Ta021	20 × 20	2297	2297	0.058	2336	1.70	2324	1.18
Ta022	20 × 20	2100	2100	0.00	2144	2.10	2112	0.57
Ta023	20 × 20	2326	2326	0.00	2364	1.63	2348	0.95
Ta024	20 × 20	2223	2223	0.00	2264	1.84	2242	0.85
Ta025	20 × 20	2291	2291	0.00	2330	1.70	2320	1.27
Ta026	20 × 20	2226	2226	0.054	2255	1.30	2249	1.03
Ta027	20 × 20	2273	2273	0.100	2303	1.32	2290	0.75
Ta028	20 × 20	2200	2200	0.038	2249	2.23	2224	1.09
Ta029	20 × 20	2237	2237	0.012	2279	1.88	2246	0.40
Ta030	20 × 20	2178	2178	0.103	2234	2.57	2192	0.64
Ta031	50 × 5	2724	2724	0.101	2735	0.40	2728	0.15
Ta032	50 × 5	2834	2834	0.115	2864	1.06	2846	0.42
Ta033	50 × 5	2621	2621	0.020	2650	1.11	2642	0.80
Ta034	50 × 5	2751	2751	0.067	2778	0.98	2762	0.40
Ta035	50 × 5	2863	2863	0.104	2887	0.84	2866	0.10
Ta036	50 × 5	2829	2829	0.075	2852	0.81	2832	0.11
Ta037	50 × 5	2725	2725	0.065	2746	0.77	2748	0.84
Ta038	50 × 5	2683	2683	0.111	2704	0.78	2690	0.26
Ta039	50 × 5	2552	2552	0.00	2586	1.33	2564	0.47
Ta040	50 × 5	2782	2782	0.00	2782	0.00	2796	0.50
Ta051	50 × 20	3875	3875	0.00	4093	5.63	3896	0.54
Ta052	50 × 20	3715	3715	0.00	3983	7.21	3746	0.83
Ta053	50 × 20	3668	3668	0.00	3911	6.62	3694	0.71
Ta054	50 × 20	3752	3752	0.00	3966	5.70	3814	1.65
Ta055	50 × 20	3635	3635	0.00	3911	7.59	3686	1.40
Ta056	50 × 20	3698	3698	0.00	3896	5.35	3722	0.65
Ta057	50 × 20	3716	3716	0.00	3998	7.59	3766	1.35
Ta058	50 × 20	3709	3709	0.00	3979	7.28	3768	1.59
Ta059	50 × 20	3765	3765	0.016	4000	6.24	3812	1.25
Ta060	50 × 20	3777	3777	0.060	4020	6.43	3826	1.30
Ta061	100 × 5	5493	5493	0.000	5505	0.22	5502	0.16
Ta062	100 × 5	5268	5268	0.015	5290	0.42	5272	0.08
Ta063	100 × 5	5175	5175	0.032	5221	0.89	5192	0.33
Ta064	100 × 5	5014	5014	0.008	5035	0.42	5020	0.12
Ta065	100 × 5	5250	5250	0.064	5280	0.57	5254	0.08
Ta066	100 × 5	5135	5135	0.044	5164	0.56	5144	0.18
Ta067	100 × 5	5246	5246	0.019	5292	0.88	5264	0.34
Ta068	100 × 5	5106	5106	0.053	5137	0.61	5114	0.16
Ta069	100 × 5	5454	5454	0.067	5506	0.95	5466	0.22
Ta070	100 × 5	5328	5328	0.066	5353	0.47	5332	0.08
Ta071	100 × 10	5770	5770	0.002	5955	3.21	5792	0.38
Ta072	100 × 10	5349	5349	0.011	5543	3.63	5368	0.36
Ta073	100 × 10	5677	5677	0.003	5823	2.57	5694	0.30
Ta074	100 × 10	5791	5791	0.013	6056	4.58	5826	0.60
Ta075	100 × 10	5468	5468	0.00	5750	5.16	5514	0.84

Instance	(nxm)	BKS	DRSO		SGA		HMSA	
			Best	PDav (%)	Best	PDav (%)	Best	PDav (%)
Ta076	100 × 10	5303	5303	0.00	5447	2.72	5324	0.40
Ta077	100 × 10	5599	5599	0.00	5747	2.64	5628	0.52
Ta078	100 × 10	5623	5623	0.00	5816	3.43	5664	0.73
Ta079	100 × 10	5875	5875	0.00	6053	3.03	5912	0.63
Ta080	100 × 10	5845	5845	0.016	5978	2.28	5892	0.80
Ta091	200 × 10	10,868	10,868	0.060	11,066	1.82	10,932	0.59
Ta092	200 × 10	10,494	10,494	0.000	10,885	3.73	10,624	1.24
Ta093	200 × 10	10,922	10,922	0.015	11,203	2.57	11,006	0.77
Ta094	200 × 10	10,889	10,889	0.032	11,036	1.35	11,024	1.24

Table 5. Comparison of Computational Results for FSSP Test Problems Using DRSO, NEH, and NEH-NGA Algorithms

Instance	(nxm)	BKS	DRSO		NEH		NEH-NGA	
			Best	PDav (%)	Best	PDav (%)	Best	PDav (%)
Ta021	20 × 20	2297	2297	0.058	2410	4.92	2297	0.00
Ta022	20 × 20	2100	2100	0.00	2150	2.38	2112	0.57
Ta023	20 × 20	2326	2326	0.00	2411	3.65	2326	0.00
Ta024	20 × 20	2223	2223	0.00	2264	1.84	2264	1.84
Ta025	20 × 20	2291	2291	0.00	2397	4.63	2305	0.61
Ta026	20 × 20	2226	2226	0.054	2349	5.53	2245	0.85
Ta027	20 × 20	2273	2273	0.100	2383	4.84	2290	0.75
Ta028	20 × 20	2200	2200	0.038	2249	2.23	2215	0.68
Ta029	20 × 20	2237	2237	0.012	2313	3.40	2248	0.49
Ta030	20 × 20	2178	2178	0.103	2277	4.55	2178	0.00
Ta031	50 × 5	2724	2724	0.101	2733	0.33	2724	0.00
Ta032	50 × 5	2834	2834	0.115	2882	1.69	2834	0.00
Ta033	50 × 5	2621	2621	0.020	2640	0.72	2630	0.34
Ta034	50 × 5	2751	2751	0.067	2787	1.31	2755	0.15
Ta035	50 × 5	2863	2863	0.104	2868	0.17	2866	0.10
Ta036	50 × 5	2829	2829	0.075	2840	0.39	2829	0.00
Ta037	50 × 5	2725	2725	0.065	2769	1.61	2736	0.40
Ta038	50 × 5	2683	2683	0.111	2707	0.89	2694	0.41
Ta039	50 × 5	2552	2552	0.00	2617	2.55	2558	0.24
Ta040	50 × 5	2782	2782	0.00	2786	0.14	2794	0.43
Ta051	50 × 20	3875	3875	0.00	4082	5.34	3880	0.13
Ta052	50 × 20	3715	3715	0.00	3921	5.55	3738	0.62
Ta053	50 × 20	3668	3668	0.00	3888	6.00	3690	0.60
Ta054	50 × 20	3752	3752	0.00	3993	6.42	3776	0.64
Ta055	50 × 20	3635	3635	0.00	3835	5.50	3673	1.05
Ta056	50 × 20	3698	3698	0.00	3914	5.84	3713	0.41
Ta057	50 × 20	3716	3716	0.00	3952	6.35	3754	1.02

Instance	(nxm)	BKS	DRSO		NEH		NEH-NGA	
			Best	PDav (%)	Best	PDav (%)	Best	PDav (%)
Ta058	50 × 20	3709	3709	0.00	3938	6.17	3709	0.00
Ta059	50 × 20	3765	3765	0.016	3952	4.97	3781	0.42
Ta060	50 × 20	3777	3777	0.060	4079	8.00	3795	0.48
Ta061	100 × 5	5493	5493	0.000	5519	0.47	5505	0.22
Ta062	100 × 5	5268	5268	0.015	5284	0.30	5268	0.00
Ta063	100 × 5	5175	5175	0.032	5219	0.85	5219	0.85
Ta064	100 × 5	5014	5014	0.008	5037	0.46	5014	0.00
Ta065	100 × 5	5250	5250	0.064	5261	0.21	5261	0.21
Ta066	100 × 5	5135	5135	0.044	5141	0.12	5141	0.12
Ta067	100 × 5	5246	5246	0.019	5266	0.38	5252	0.11
Ta068	100 × 5	5106	5106	0.053	5107	0.02	5106	0.00
Ta069	100 × 5	5454	5454	0.067	5500	0.84	5474	0.37
Ta070	100 × 5	5328	5328	0.066	5346	0.34	5346	0.34
Ta071	100 × 10	5770	5770	0.002	5846	1.32	5780	0.17
Ta072	100 × 10	5349	5349	0.011	5453	1.94	5358	0.17
Ta073	100 × 10	5677	5677	0.003	5781	1.83	5700	0.41
Ta074	100 × 10	5791	5791	0.013	5942	2.61	5833	0.73
Ta075	100 × 10	5468	5468	0.00	5679	3.86	5509	0.75
Ta076	100 × 10	5303	5303	0.00	5375	1.36	5319	0.30
Ta077	100 × 10	5599	5599	0.00	5723	2.21	5644	0.80
Ta078	100 × 10	5623	5623	0.00	5737	2.03	5668	0.80
Ta079	100 × 10	5875	5875	0.00	5983	1.84	5896	0.36
Ta080	100 × 10	5845	5845	0.016	5903	0.99	5890	0.77
Ta091	200 × 10	10,868	10,868	0.060	10,942	0.68	10,968	0.92
Ta092	200 × 10	10,494	10,494	0.000	10,735	2.30	10,594	0.95
Ta093	200 × 10	10,922	10,922	0.015	11,027	0.96	10,992	0.64
Ta094	200 × 10	10,889	10,889	0.032	11,057	1.54	10,984	0.87
Ta095	200 × 10	10,524	10,524	0.008	10,684	1.52	10,565	0.39

Table 6. Comparison of Computational Results for FSSP Test Problems Using DRSO, SSO, SCE-OBL, CLS-BFO, and ACGA Algorithms

Instance	(n*m)	BKS	DRSO	SSO	SCE-OBL	CLS-BFO	ACGA
Rec1	20x5	1245	1245	1247	1249	1249	1249
Rec3	20x5	1109	1109	1109	1111	1111	1109
Rec5	20x5	1242	1242	1245	1245	1245	1245
Rec7	20x10	1566	1566	1566	1584	1584	1566
Rec9	20x10	1537	1537	1537	1545	1545	1537
Rec11	20x10	1431	1431	1431	1431	1449	1431
Rec13	20x15	1930	1930	1935	1963	1968	1935
Rec15	20x15	1950	1950	1968	1993	1993	1950
Rec17	20x15	1902	1902	1923	1944	1954	1911

Instance	(n*m)	BKS	DRSO	SSO	SCE-OBL	CLS-BFO	ACGA
Rec19	30x10	2093	2093	2117	2156	2139	2099
Rec21	30x10	2017	2017	2017	2064	2059	2046
Rec23	30x10	2011	2011	2030	2067	2073	2021
Rec25	30x15	2513	2513	2566	2584	2638	2545
Rec27	30x15	2373	2373	2397	2445	2443	2396
Rec29	30x15	2287	2287	2333	2364	2408	2304
Rec31	50x10	3045	3045	3104	3179	3180	3105
Rec33	50x10	3114	3114	3118	3154	3187	3140
Rec35	50x10	3277	3277	3277	3281	3292	3277
Rec37	75x20	4890	4890	5096	5327	5422	5193
Rec39	75x20	5043	5043	5185	5391	5465	5276
Rec41	75x20	4910	4910	5135	5334	5436	5208

Table 7. Comparison of Computational Results for FSSP Test Problems Using DRSO and CSO Algorithms

Instances	DRSO			CSO		
	Best	Average	PDAV	Best	Average	PDAV
Ta001	1278	1279,02	0.08	1278	1278	0.00
Ta011	1582	1584,05	0.13	1586	1603.8	1.37
Ta015	1419	1419	0.00	1426	1443.9	1.75
Ta021	2297	2301,36	0.19	2308	2319.9	0.99
Ta025	2291	2291	0.00	2312	2318.8	1.21
Ta031	2724	2726,75	0.101	2724	2725	0.04
Ta035	2863	2865,98	0.104	2863	2863.8	0.03
Ta040	2782	2782	0.00	2782	2782	0.00
Ta041	2991	2991	0.00	3063	3074.9	2.81
Ta045	2976	2976	0.00	3035	3065.7	3.01
Ta051	3875	3875	0.00	3968	3978.6	3.42
Ta055	3635	3635	0.00	3750	3772.5	6.17
Ta061	5493	5493	0.000	5493	5493.8	0.037
Ta065	5250	5253,36	0.064	5255	5255	0.09
Ta071	5770	5770,11	0.002	5791	5802	0.55
Ta075	5468	5468	0.00	5512	5548.8	1.49

6. Comparison

In this section, we will proceed to the comparison of the DRSO algorithm with other metaheuristics based on the data provided by the authors of the compared methods. The objective is to evaluate and analyze the performance of these algorithms to determine the relative efficiency of DRSO.

The algorithms are evaluated according to three main criteria: the best solution found (Best), the average solution found (Average), and the percentage of deviation from the best-known solution (PDav).

For each comparison, we will apply a parametric or non-parametric test, depending on the size of the samples studied and the data provided.

The statistical tests used are the Holm-Šídák multiple comparison test and the Wilcoxon test.

The Holm-Šídák test is a multiple comparison method that controls the type I error rate when examining several hypotheses simultaneously.

The Wilcoxon test is a non-parametric test that compares the medians of two samples to determine if they are from the same population.

Each comparison will be illustrated by a graph showing the PDav comparison curve or the best value obtained, to justify the performance of the DRSO algorithm, as illustrated in the five Figures (4-9).

6.1.1. Comparison between DRSO, IIGA, DPSOVND, TMIIG, and DWWO

The results in Table 2 show that the DRSO algorithm reached the optimum for all instances (i.e. 100%) with an average close to or equal to the optimum in most cases. In contrast, the other algorithms such as IIGA, DPSOVND, TMIIG, and DWWO failed to find the optimum for all instances (0 out of 65). The average results of these algorithms were also very high compared to the optimum found. Thus, the performance of DRSO is significantly better than the other algorithms.

In the Figure 4, it can be observed that the curve of DRSO is significantly smaller than the other curves, indicating that DRSO is more stable and has better overall performance than the other algorithms.

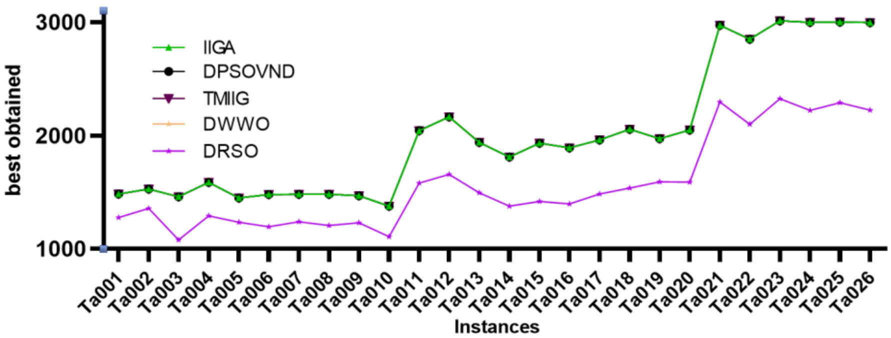

Figure 4. Comparison of the best value obtained by DRSO, IIGA, DPSOVND, TMIIG, and DWWO

In Table 8, the results of the Holm-Šídák multiple comparison test comparing the performance of the DRSO method with that of the IIGA, DPSOVND, TMIIG, and DWWO methods are displayed. The test results indicate that the differences in performance between the DRSO method and the other methods (IIGA, DPSOVND, TMIIG, and DWWO) are statistically significant.

Specifically, the negative mean difference for each comparison suggests that DRSO performs better than the other methods. Additionally, the adjusted P values for all comparisons are less than 0.0001, indicating a very high level of significance.

Table 8. Holm-Šídák Multiple Comparisons Test Results for DRSO and IIGA, DPSOVND, TMIIG, and DWWO

Test	Mean DIFF,	Below threshold?	Summary	Adjusted P value
DRSO VS. IIGA	-936,8	YES	****	<0,0001
DRSO VS. DPSOVND	-922,5	YES	****	<0,0001
DRSO VS. TMIIG	-914,6	YES	****	<0,0001
DRSO VS. DWWO	-917,2	YES	****	<0,0001

In detail, the mean difference between DRSO and IIGA is -936.8, between DRSO and DPSOVND, is -922.5, between DRSO and TMIIG, is -914.6, and between DRSO and DWWO is -917.2. In all of these comparisons, the DRSO method shows superior performance, as indicated by the four stars (****) in the abstract, which indicate a very high level of significance.

6.1.2. Comparison between DRSO, BAT, and TLBO

From Table 3, the comparison of the performance of the DRSO, BAT, and TLBO methods in solving the instances of the problem Ta shows significant differences in the percentage of success in finding the best solution equal to the best-known solution (BKS) as well as in the percentage of average deviation (PDav).

DRSO appears to be the best-performing method for finding the best solution equal to BKS for all instances. Conversely, BAT performs less well, reaching the best solution equal to BKS only 16.67% of the time. TLBO performs the worst of the three methods, with 8.33%.

In terms of percent average deviation (PDav), DRSO generally has a low deviation, indicating that the performance of this method is close to the best-known solution. In the majority of cases, BAT exhibits a higher percentage of mean deviation than DRSO, suggesting lower accuracy for this method. Similarly, TLBO has a higher average deviation percentage than DRSO in many cases and sometimes even higher than BAT, indicating that its performance is less accurate than the other two methods.

In addition, Figure 5 shows the comparison of the performance curves of the algorithms. The curve of DRSO is significantly smaller than those of BAT and TLBO, indicating better stability and overall superior performance for DRSO compared to the other algorithms.

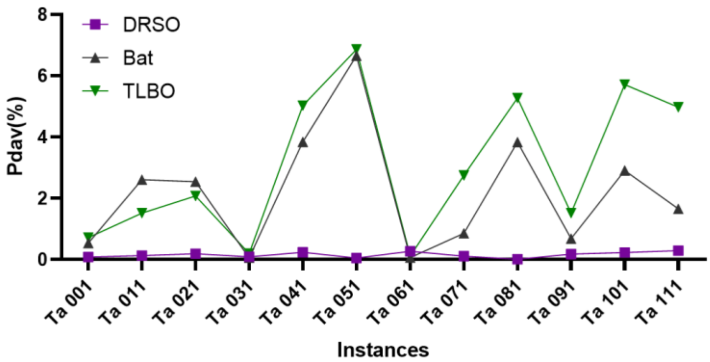

Figure 5. Comparison of the PDav(%) value obtained by DRSO, BAT, and TLBO

In Table 9, the results of the Holm-Šídák test indicate a significant difference in performance between DRSO and BAT, with a mean difference of -120.9 and an adjusted P value of 0.0240. The negative difference suggests that DRSO performs better than BAT. This comparison has a level of significance, as indicated by the (*) in the summary column.

In contrast, the comparison between DRSO and TLBO does not show a significant difference in performance. The mean difference is -218.0 and the adjusted P value is 0.0515, slightly above the 0.05 significance level. In this comparison, the summary indicates "ns" (not significant), which means that there is insufficient evidence to conclude that the performance of DRSO is significantly different from that of TLBO

Table 9. Holm-Šídák Multiple Comparisons Test Results for DRSO and IIGA, DPSOVND, TMIIG, and DWWO

Test	Mean DIFF,	Below threshold?	Summary	Adjusted P value
DRSO VS. BAT	-120,9	YES	*	0,0240
DRSO VS. TLBO	-218,0	NO	NS	0,0515

6.1.3. Comparison between DRSO, SGA, and HMSA

From Table 4, the comparison of the performance of the DRSO, SGA, and HMSA methods in solving the instances of the problem Ta shows significant differences in their ability to find the best solution equal to the best-known solution (BKS) as well as in the percentage of average deviation (PDav).

DRSO appears to be the best-performing method for finding the best solution equal to BKS for all instances. SGA and HMSA, on the other hand, are less effective in obtaining the best solution equal to BKS, with HMSA generally outperforming SGA.

In terms of percent average deviation (PDav), DRSO consistently shows a low deviation, indicating that the performance of this method is close to the best-known solution. In most cases, SGA exhibits a higher percentage of mean deviation than DRSO, suggesting lower accuracy for this method. Similarly, HMSA often has a higher average deviation percentage than DRSO, indicating that its performance is less accurate than DRSO, but it is generally more accurate than SGA.

Figure 6 illustrates the consistency of DRSO's performance compared to SGA and HMSA using a graphical representation. The curve highlights the low deviation and higher accuracy of DRSO, while SGA and HMSA show higher average deviation percentages. Therefore, this Figure supports the claim that DRSO is a more reliable and accurate algorithm than SGA and HMSA.

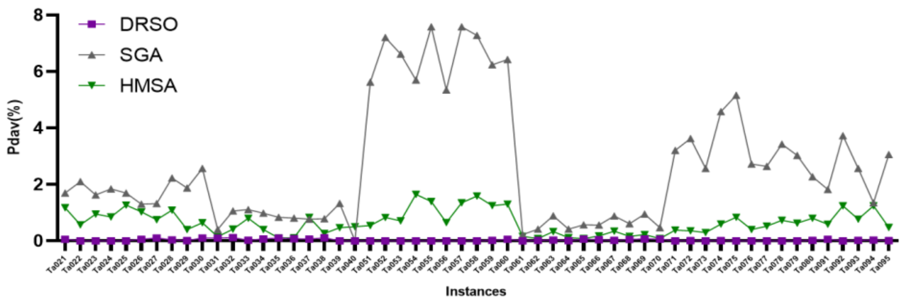

Figure 6. Comparison of the PDav(%) value obtained by DRSO, SGA, and HMSA

The results of the Holm-Šídák multiple comparison test in Table 10 show that the differences in performance between DRSO and the other two methods (SGA and HMSA) are statistically significant. The negative mean difference suggests that DRSO performs better than SGA and HMSA, with an adjusted P value of less than 0.0001. The four stars (****) in the abstract indicate a very high level of significance for these comparisons.

Table 10. Holm-Šídák Multiple Comparisons Test Results for DRSO, SGA, and HMSA

Test	Mean DIFF,	Below threshold?	Summary	Adjusted P value
DRSO VS. SGA	-116,2	YES	****	<0,0001
DRSO VS. HMSA	-28,59	YES	****	<0,0001

6.1.4. Comparison between DRSO, NEH, and NEH-NGA

Comparing the results of the three methods, DRSO, NEH, and NEH-NGA in Table 5, reveals that the DRSO method performs best in solving the scheduling. DRSO finds the best solution, equal to the best-known solution (BKS), for 42 out of 54 instances, resulting in a success rate of approximately 77.78%. In addition, this method has a very low percentage average deviation (PDav) of 0.037%, indicating higher accuracy than the other methods.

In comparison, the NEH method fails to find the best solution for any of the 54 instances, with a success percentage of 0%. Its average PDav is 2.82%, which shows a significant difference from BKS.

The NEH-NGA method, on the other hand, succeeds in finding the best solution for 12 of the 54 instances, with a success rate of about 22.22%. Its average PDav is 0.43%, which is a relatively small difference on average, but still higher than that of the DRSO method.

Figure 7 shows a comparison of the PDav(%) values obtained by DRSO, NEH, and NEH-NGA. The Figure shows that the DRSO method has an exceptionally low percent mean deviation (PDav), which means higher accuracy than the other methods examined.

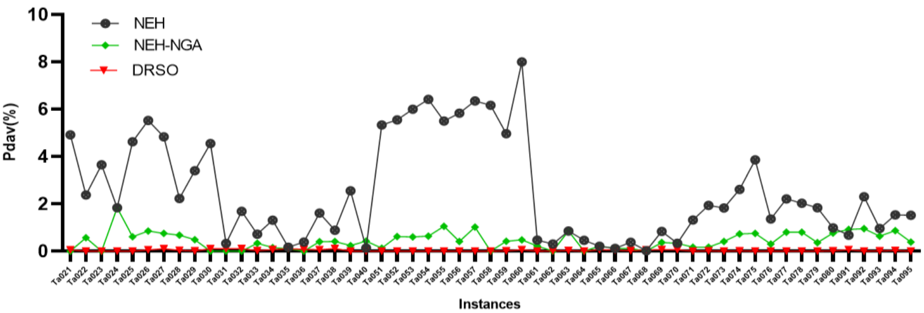

Figure 7. Comparison of the PDav(%) value obtained by DRSO, NEH, NEH-NGA

The results of the Holm-Šídák multiple comparison tests, presented in Table 11, indicate statistically significant differences in performance between DRSO and the other two methods (NEH and NEH-NGA). The negative mean differences suggest that DRSO performs better than NEH and NEH-NGA. Adjusted P values less than 0.0001 for both comparisons, represented by the four stars (****), denote an exceptionally high level of significance.

Table 11. Holm-Šídák Multiple Comparisons Test Results for DRSO, NEH, and NEH-NGA

Test	Mean Diff,	Below threshold?	Summary	Adjusted P Value
DRSO vs. NEH	-114,3	Yes	****	<0,0001
DRSO vs. NEH-NGA	-32,43	Yes	****	<0,0001

6.1.5. Comparison between DRSO, SSO, SCE-OBL, CLS-BFO, and ACGA

Table 6 compares the performance of the DRSO algorithm to four other optimization methods: SSO, SCE-OBL, CLS-BFO, and ACGA, on 21 instances of a problem characterized by "nxm" matrices. The evaluation criterion for these algorithms is their capacity to achieve the best-known solution (BKS) for each instance. Remarkably, DRSO consistently reaches the BKS in all 21 instances, boasting a 100% success rate. In contrast, the other algorithms exhibit varying success levels in attaining the BKS, with SSO accomplishing it in merely 7 out of 21 instances, while the other three methods also fall short of DRSO's performance.

As indicated in Table 12, out of the 21 instances, DRSO surpasses the SSO algorithm in 11 instances (52.38%), SCE-OBL in 15 instances (71.43%), CLS-BFO in 17 instances (80.95%), and ACGA in 16 instances (76.19%).

Table 12. The percentage of instances where DRSO outperformed other algorithms

ALGORITHM	DRSO OUTPERFORMS (%)
SSO	52.38%
SCE-OBL	71.43%
CLS-BFO	80.95%
ACGA	76.19%

Figure 8 shows the comparison of the best value obtained by DRSO, SSO, SCE, CLS-BFO, and ACGA. The curve for DRSO is significantly lower. This indicates that DRSO systematically obtains better results in terms of the best value obtained, which underlines its superior performance and efficiency compared to the other algorithms.

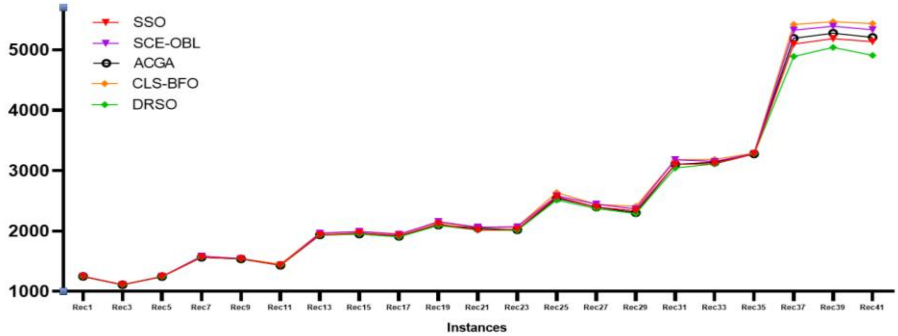

Figure 8. Comparison of the best value obtained by DRSO, SSO, SCE, CLS-BFO, and ACGA

Holm-Šídák's multiple comparison analysis in Table 13 is used to compare the performance of DRSO against the other methods. The results show that DRSO performs significantly better than SSO, SCE-OBL, CLS-BFO, and ACGA in all cases studied, with mean differences of 40.52, 91.71, 112.1, and 50.38, respectively. Holm-Šídák adjustments were applied to control for type I errors. Adjusted p values were calculated for each comparison and were all less than 0.05, indicating a significant difference between the performance of DRSO and the other methods.

Table 13. The percentage of instances where DRSO outperformed other algorithms

TEST	MEAN DIFF,	BELOW THRESHOLD?	SUMMARY	ADJUSTED P VALUE
DRSO VS. SSO	-40,52	YES	*	0,0232
DRSO VS. SCE-OBL	-91,71	YES	*	0,0219
DRSO VS. CLS-BFO	-112,1	YES	*	0,0219
DRSO VS. ACGA	-50,38	YES	*	0,0276

6.1.6. Comparison between DRSO, and CSO

Table 7 compares the computational results for the FSSP test problems using the DRSO and CSO algorithms, highlighting the performance of each algorithm in terms of Best, Average and PDAV values.

For the DRSO algorithm, the Best and Average values are either identical or very close, indicating consistent and stable convergence to the BKS. The PDAV values for the DRSO algorithm are also very low, confirming its stability.

In contrast, the performance of the CSO algorithm varies from instance to instance. In some instances, the best values are equal to those obtained by DRSO (e.g., Ta001, Ta031, Ta035, Ta040, Ta061), while in others, the best values are higher than those of DRSO (e.g., Ta011, Ta015, Ta021, Ta025, Ta041, Ta045, Ta051, Ta055, Ta065, Ta071, Ta075). The average values are generally higher than the optimal values, suggesting less stability in the convergence of the CSO algorithm. The PDAV values for CSO are also higher than those for DRSO, reflecting the more variable performance of the CSO algorithm.

Figure 9 illustrates the stability of the DRSO algorithm, which consistently converges to the BKS. The low "PDAV" values presented in the Figure confirm the efficiency and stability of the algorithm.

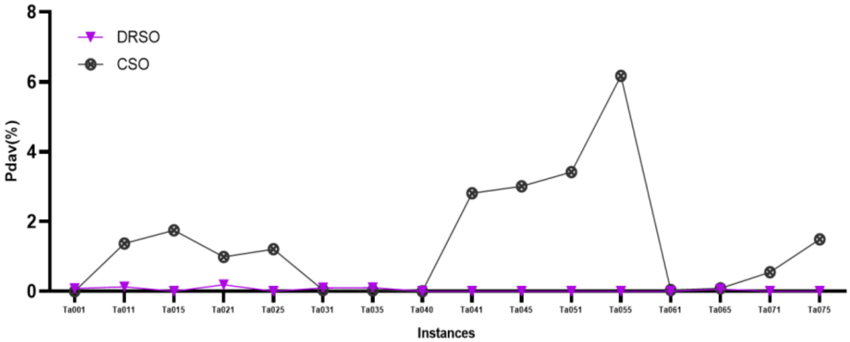

Figure 9. Comparison of the Pdav(%) value obtained by DRSO, and CSO

Based on the Wilcoxon test information provided in Table 13, to compare DRSO and CSO. The results of the test indicate a P value of 0.0010 and a summary P value of ***, which means that the two groups are significantly different with a significance level of P < 0.05.

Table 14. Wilcoxon signed rank Comparisons Test Results for DRSO and CSO

Wilcoxon signed-rank test	value
P value	0,0010
P value summary	***
Significantly different (P < 0.05)?	Yes

6.2. Evaluating DRSO Performance Using Analysis and Friedman Test

The Friedman test with an alpha of 0.05 and a 95% confidence interval can also be used to justify the performance of the DRSO optimization algorithm. The Friedman test is a statistical test that measures the significance of the difference between two data sets. By setting the alpha to 0.05 and the confidence interval to 95%, we can determine whether the difference in performance between DRSO and the other algorithms is statistically significant.

If the Friedman test reveals that the difference in performance between DRSO and the other algorithms is statistically significant with a p-value less than 0.05, we can conclude that the performance of DRSO is significantly better than the other algorithms. This means that we can be 95% sure that the observed differences in performance are not due to chance or random variation, but rather to the inherent superiority of the DRSO algorithm.

Based on the results of the Friedman test with an alpha of 0.05 and a 95% confidence interval, as presented in Table 9, it appears that DRSO outperforms the other optimization algorithms in terms of finding the optimal solution. The multiple comparisons test shows that DRSO has a significantly lower rank sum difference than BAT, TLBO, TMIIG, DWWO, BAT, TLBO, SGA, HMSA, NEH, NEH-NGA, CLS-BFO, and ACGA. This is indicated by "Yes" in the "Significant?" column, and the adjusted p-value is less than 0.0001 for all these comparisons.

However, the results indicate that DRSO does not have a significantly lower rank sum difference than SSO or SCE-OBL. The "No" in the "Significant?" column and the adjusted p-value are greater than 0.9999 for SSO and 0.0873 for SCE-FBL.

These results, as shown in Table 9, suggest that DRSO is a highly efficient optimization algorithm compared to the other algorithms tested. It consistently outperformed the other algorithms in finding the optimal solution with accuracy.

Table 9. The Friedman test for the difference between DRSO and the other algorithms

Test	Rank sum diff	Significant?	P-Value
DRSO vs. IIGA	-190	Yes	<0,0001
DRSO vs. DPSOVND	-165,5	Yes	<0,0001
DRSO vs. TMIIG	-77	Yes	<0,0001
DRSO vs. DWWO	-92,5	Yes	<0,0001
DRSO vs. BAT	-11	Yes	0,0495
DRSO vs. TLBO	-16	Yes	0,0022
DRSO vs. SGA	-107	Yes	<0,0001
DRSO vs. HMSA	-56,5	Yes	<0,0001
DRSO vs. NEH	-100,5	Yes	<0,0001
DRSO vs. NEH-NGA	-49,5	Yes	<0,0001
DRSO vs. SSO	0	No	>0,9999
DRSO vs. SCE-OBL	-23,5	No	0,0873
DRSO vs. CLS-BFO	-53	Yes	<0,0001
DRSO vs. ACGA	-63,5	Yes	<0,0001
DRSO vs. CSO	-83,5	Yes	<0,0001

7. Conclusion

In summary, the utilization of discrete rat swarm optimization in manufacturing systems shows significant promise for improving efficiency and productivity. Implementing this approach could contribute to considerable advancements in manufacturing processes, resulting in more streamlined and cost-effective operations.

The implementation of discrete rat swarm optimization has demonstrated its efficacy in addressing the flow shop-scheduling problem, indicating its potential to enhance the efficiency of manufacturing systems. The ability of this method to identify optimal solutions with a high degree of accuracy positions it as a valuable tool for boosting manufacturing process productivity.

When compared to other optimization algorithms, such as BAT, TLBO, TMIIG, DWWO, SGA, HMSA, NEH, NEH-NGA, CLS-BFO, and ACGA, discrete rat swarm optimization consistently outperforms these techniques in obtaining the optimal solution. This evidence underscores the superiority of this approach for solving complex optimization challenges.

The integration of discrete rat swarm optimization within manufacturing systems offers immense potential for substantially enhancing efficiency and productivity. By adopting this approach, significant advancements in manufacturing processes can be realized, ultimately leading to more streamlined and cost-effective operations.

In future work, we will focus on several key aspects to advance the current state of research. First, we will continue to refine the performance of discrete rat swarm optimization in the context of the shop floor scheduling problem to improve its efficiency and impact. In addition, we will explore the potential applications of this optimization technique in other optimization tasks, thus expanding its scope and influence in various domains. In addition, we will develop hybrid optimization algorithms that combine the strengths of rat swarm optimization with those of other optimization techniques, which could lead to significant improvements in the overall efficiency of this method.

Author contributions: Research problem, M.R., T.M. and I.M.; Conceptualization, M.R., T.M., and I.M.; Methodology, M.R., T.M., and I.M.; Formal analysis, M.R., T.M.; Resources, M.R., I.M.; Original drafting, M.R. and T.M.; Reviewing and editing, M.R., T.M., I.M; Project administration, M.R., T.M., IM; Supervision, M.R., I.M.

Funding: This research received no external funding.

Data Availability Statement: The data used to support the findings of this study are included within the article.

Conflicts of 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.

References

Ab Wahab, M. N., Nefti-Meziani, S., & Atyabi, A. (2015). A Comprehensive Review of Swarm Optimization Algorithms. PLOS ONE, 10(5), e0122827. https://doi.org/10.1371/journal.pone.0122827

Alaliyat, S., Yndestad, H., & Sanfilippo, F. (2014). Optimisation Of Boids Swarm Model Based On Genetic Algorithm And Particle Swarm Optimisation Algorithm (Comparative Study). ECMS 2014 Proceedings Edited by: Flaminio Squazzoni, Fabio Baronio, Claudia Archetti, Marco Castellani, 643–650. https://doi.org/10.7148/2014-0643

Babaee Tirkolaee, E., Goli, A., & Weber, G.-W. (2020). Fuzzy Mathematical Programming and Self-Adaptive Artificial Fish Swarm Algorithm for Just-in-Time Energy-Aware Flow Shop Scheduling Problem With Outsourcing Option. IEEE Transactions on Fuzzy Systems, 28(11), 2772–2783. https://doi.org/10.1109/TFUZZ.2020.2998174

Bellabai, J. R., Leela, B. N. M., & Kennedy, S. M. R. (2022). Testing the Performance of Bat-Algorithm for Permutation Flow Shop Scheduling Problems with Makespan Minimization. Brazilian Archives of Biology and Technology, 65. https://doi.org/10.1590/1678-4324-2022210840

Blum, C. (2005). Ant colony optimization: Introduction and recent trends. Physics of Life Reviews, 2(4), 353–373.

Ding, J.-Y., Song, S., Gupta, J. N. D., Zhang, R., Chiong, R., & Wu, C. (2015). An improved iterated greedy algorithm with a Tabu-based reconstruction strategy for the no-wait flowshop scheduling problem. Applied Soft Computing, 30, 604–613.

Huang, Y.-M., & Lin, J.-C. (2011). A new bee colony optimization algorithm with idle-time-based filtering scheme for open shop-scheduling problems. Expert Systems with Applications, 38(5), 5438–5447.

Kurdi, M. (2021). Application of Social Spider Optimization for Permutation Flow Shop Scheduling Problem. Journal of Soft Computing and Artificial Intelligence, 2(2), 85–97.

Liang, Z., Zhong, P., Liu, M., Zhang, C., & Zhang, Z. (2022). A computational efficient optimization of flow shop scheduling problems. Scientific Reports, 12(1), 845. https://doi.org/10.1038/s41598-022-04887-8

Li, X., & Yin, M. (2013). A hybrid cuckoo search via Lévy flights for the permutation flow shop scheduling problem. International Journal of Production Research, 51(16), 4732–4754.

Marichelvam, M. K., Tosun, Ö., & Geetha, M. (2017). Hybrid monkey search algorithm for flow shop scheduling problem under makespan and total flow time. Applied Soft Computing, 55, 82–92. https://doi.org/10.1016/j.asoc.2017.02.003

Mzili, T., Riffi, M. E., Mzili, I., & Dhiman, G. (2022). A novel discrete Rat swarm optimization (DRSO) algorithm for solving the traveling salesman problem. Decision Making: Applications in Management and Engineering, 5(2), 287–299.

Pan, Q.-K., Wang, L., & Zhao, B.-H. (2008). An improved iterated greedy algorithm for the no-wait flow shop scheduling problem with makespan criterion. The International Journal of Advanced Manufacturing Technology, 38(7–8), 778–786.

Reza Hejazi *, S., & Saghafian, S. (2005). Flowshop-scheduling problems with makespan criterion: a review. International Journal of Production Research, 43(14), 2895–2929.

Smutnicki, C., Pempera, J., Bocewicz, G., & Banaszak, Z. (2022). Cyclic flow-shop scheduling with no-wait constraints and missing operations. European Journal of Operational Research, 302(1), 39–49.

Tanaev, V. S., Gordon, V. S., & Shafransky, Y. M. (1994). NP-Hard Problems. In Scheduling Theory. Single-Stage Systems (pp. 253–311). Springer Netherlands. https://doi.org/10.1007/978-94-011-1190-4_5

Wang, J., & Magron, V. (2022). Exploiting Sparsity in Complex Polynomial Optimization. Journal of Optimization Theory and Applications, 192(1), 335–359.

Wu, X., Yan, X., & Wang, L. (2022). Optimizing job release and scheduling jointly in a reentrant hybrid flow shop. Expert Systems with Applications, 209, 118278. https://doi.org/10.1016/j.eswa.2022.118278

Zhang, J., Zhang, C., & Liang, S. (2010). The circular discrete particle swarm optimization algorithm for flow shop scheduling problem. Expert Systems with Applications, 37(8), 5827–5834.

Zheng, C., Xing, J., Wang, Z., Qin, X., Eynard, B., Li, J., Bai, J., & Zhang, Y. (2022). Knowledge-based program generation approach for robotic manufacturing systems. Robotics and Computer-Integrated Manufacturing, 73, 102242. https://doi.org/10.1016/j.rcim.2021.102242