Proceedings of the International Conference on
Formal Methods and Foundations of Artificial Intelligence
Eszterházy Károly Catholic University
Eger, Hungary, June 5–7, 2025
pp. 4–13 DOI: 10.17048/fmfai.2025.4

Efficiency analysis comparison of the Particle Swarm Optimization and Tabu Search in Flow Shop Scheduling Problem*

Anita Agárdi

Institute of Informatics, University of Miskolc
anita.agardi@uni-miskolc.hu

Abstract. The paper investigates the efficiency of two metaheuristic algorithms, the Particle Swarm Optimization and the Tabu Search, on the Flow Shop Scheduling Problem. Particle Swarm Optimization is a population algorithm. The algorithm maintains a population of solutions. It improves the population during the iteration. Tabu Search improves a single possible solution. The paper presents the efficiency of the algorithms on a benchmark dataset and compares it with results published by other researchers.

1. Introduction

Industry 4.0 [4] is also known as the fourth industrial revolution. It has fundamentally changed manufacturing and industrial processes. It integrates digitalization, automation and artificial intelligence. This new approach enables production systems to become smarter. Machines and equipment no longer perform only traditional, pre-programmed tasks. It can process data in real time and perform analyses. It also can make autonomous decisions to increase efficiency. The principles of Industry 4.0 is the closer connection of information technology and industrial processes. Every element of the production chain can be continuously monitored, analysed and optimised. Internet of Things (IoT), cloud computing, robotics, augmented reality and data analytics are important in this process.

Production scheduling [7] aims to efficiently allocate and utilize available resources, such as machinery, labor, and materials. Production scheduling involves

*Supported by the University Research Scholarship Program of the Ministry for Culture and Innovation from the source of the National Research, Development and Innovation Fund.

scheduling production orders, setting priorities etc. This process also takes into account production capacities, machine maintenance needs and order deadlines. The aim of production scheduling is the downtime minimization, reducing unnecessary inventory accumulation.

Metaheuristic algorithms [1] are optimization methods. The aim is to find near-optimal solutions to complex problems. These algorithms do not guarantee finding the globally optimal solution. The algorithms try to navigate the solution space efficiently. They usually combine random search elements with structured, heuristic methods. They are able to avoid local optima and find solution near the global optimum. For example, Genetic Algorithm [13], Simulated Annealing [5], Tabu Search [2], Particle Swarm Optimization [14], etc.

2. Materials and methods

2.1. Flow Shop Scheduling

Flow Shop Scheduling (FSS) [6] is a production scheduling problem. During the problem, products or workpieces must pass through different work processes or machines. The objective function of the problem is minimizing production time, downtime, and/or delays. All workpieces pass through the same machines and operations. The processing times assigned to each machine can be different. Establishing the correct sequence and schedule can be very difficult. Formally, a Flow Shop Scheduling problem can be defined as follows:

Consider a set of n jobs $J = \{J_1, J_2, \ldots, J_n\}$ and a set of m machines $M = \{M_1, M_2, \ldots, M_m\}$. Each job J_i consists of a sequence of m operations, one for each machine, with processing time $p_{i,j}$ on machine M_j. The objective is to find a permutation π of the jobs that minimizes a given criterion, such as the makespan, total completion time, or total tardiness.

Mathematically, the classical Flow Shop Scheduling problem with makespan minimization can be written in a following way:

$$C_{\max} = \max_{i=1,\ldots,n} \{C_{i,m}\},$$

where $C_{i,j}$ is the completion time of job J_i on machine M_j, calculated recursively as:

$$\begin{cases} C_{1,1} = p_{1,1}, & \\ C_{i,1} = C_{i-1,1} + p_{i,1}, & i = 2, \ldots, n, \\ C_{1,j} = C_{1,j-1} + p_{1,j}, & j = 2, \ldots, m, \\ C_{i,j} = \max(C_{i-1,j}, C_{i,j-1}) + p_{i,j}, & i = 2, \ldots, n,\ j = 2, \ldots, m. \end{cases}$$

Typical examples of Flow Shop Scheduling applications are:

- Automotive assembly: Parts are assembled in a specific order.
- Electronics production: Testing, assembling, and quality control of components.

- Metalworking: Cutting, welding, and grinding of metals.

2.2. Particle Swarm Optimization

Particle Swarm Optimization (PSO) [3] is a population-based optimization algorithm. The algorithm is inspired by the movement of particles. It was developed by James Kennedy and Russell Eberhart in 1995. Particles represent potential solutions that move through the search space based on their own experience and that of the swarm. Steps of the algorithm:

1. **Particle creation:** Each particle represents a possible solution. The elements of the population can be randomly generated or can be the results of a construction algorithm.

2. **Velocity and position update:** Particles get new velocity and position in each iteration. Velocity is updated based on the following formula:

$$\vec{v}_i(t+1) = w \cdot \vec{v}_i(t) + c_1 \cdot r_1 \cdot (\vec{pbest}_i - \vec{x}_i(t)) + c_2 \cdot r_2 \cdot (\vec{gbest} - \vec{x}_i(t))$$

where:

$\vec{v_i}(t)$: velocity of particle i in iteration t

$\vec{x_i}(t)$: position of particle i in iteration t

w: inertia weight

c_1, c_2: learning factors that influence attraction towards $\vec{pbest}_i$ and $\vec{gbest}$

$\vec{pbest}_i$: best location of particle i

$\vec{gbest}$: global best solution

r_1 and r_2: are random numbers between $[0, 1]$. They are independently regenerated for each particle in every iteration. Their role is to provide a random weighting for the particle's movement. r_1 scales the "cognitive" component, pulling the particle towards its own best position ($\vec{pbest}_i$). r_2 scales the "social" component, pulling the particle towards the global best position ($\vec{gbest}$).

The position of particles is updated with the following formula:

$$\vec{x_i}(t+1) = \vec{x_i}(t) + \vec{v_i}(t+1)$$

3. **Termination condition:** The algorithm stops if the termination condition is met. The termination condition can be a certain number of iterations, convergence, or a fixed runtime.

Particle Swarm Optimization was originally developed for continuous tasks, but the Flow Shop Scheduling task is a discrete problem, so the algorithm needs to be discretized. The discretization was based on the article [10], which solves the Traveling Salesman Problem.

Algorithm 1: Discrete Particle Swarm Optimization for Flow Shop Scheduling.

Input: Flow Shop Scheduling problem instance
Output: Best found solution

1. Create the initial particles. The particles' positions $\vec{x}_i(t)$ represent permutations in the Flow Shop Scheduling problem. In this case, they are generated randomly.

2. Initialize the particles' velocities $\vec{v}_i(t)$. This is a Basic Swap Sequence value [10], since the problem is discrete.

while *termination criteria is not met* **do**

  2.a Compute the global best particle.

  foreach *particle* **do**

    3. Compute the particle velocity using the following formula:

$$\vec{v}_i(t+1) = \vec{v}_i(t) \oplus c_1 r_1 (\vec{pbest}_i - \vec{x}_i(t)) \oplus c_2 r_2 (\vec{gbest} - \vec{x}_i(t))$$

    Here, the Basic Swap Sequence [10] represents the sequence of swaps between:

    – the best position of the particle $\vec{pbest}_i$ and its current position (both are permutations) ie. $\vec{pbest}_i - \vec{x}_i(t)$, and

    – the global best $\vec{gbest}$ and the particle's current position ie. $\vec{gbest} - \vec{x}_i(t)$.

    The operator $\oplus$ denotes performing swaps between permutations (adding Basic Swap Sequence values).

    4. Determine the current particle's new position using the following formula:

$$\vec{x}_i(t+1) = \vec{x}_i(t) + \vec{v}_i(t+1)$$

    This means performing the swaps defined by the Basic Swap Sequence on the permutation.

    5. Determine the fitness value of $\vec{x}_i(t+1)$;

2.3. Tabu Search

Tabu Search (TS) [12] is a local search procedure. The algorithm maintains a tabu list to move from a local optimum to a global optimum.

The steps of the algorithm are the followings:

- **Initial solution:** Taking an initial solution (it is either randomly generated or constructed using some construction technique). This solution will be the current solution. In the case of Flow Shop Scheduling Problem each solution means a randomly generated permutation.

- **Neighborhood search:** Searching for the neighbors of the current solution using a neighborhood operator. In the case of Flow Shop Scheduling Problem, the swap operator is used for this.

- **Tabu list:** The tabu list stores those solutions that cannot be re-selected (already visited solutions).
- **Recording best solution:** The algorithm continuously monitors the best solution found so far.
- **Deleting individual elements of the tabu list:** If the tabu list is full, the older solutions are deleted.
- **Termination condition:** The termination condition can be a certain number of iterations, convergence, or a predefined runtime.

3. Test results

This section contains the test results. First, the results of Particle Swarm Optimization, then the Tabu Search test runs are presented.

Separate tables are provided for both PSO and TS algorithms, showing the maximum, average, and minimum fitness values of the runs for each Taillard data set (Table 1, Table 3). The article also examines the fitness values of the proposed algorithms in comparison with the already published data. Table 2 and Table 4 contain the fitness values of the PSO or TS and and how much better the TS or PSO algorithm is than other algorithms published in the literature. If the percentage is positive, then the PSO or TS algorithm is better than the given comparison algorithm. The comparisons were created with the published results of the following algorithms [8, 11]:

- HMM-PFA – Hidden Markov Model based Particle Filter Algorithm
- HGA – Hybrid Genetic Algorithm
- IIGA – Improved Invasive Weed Optimization Algorithm
- DSOMA – Differential Search Optimization Method Algorithm
- HGSA – Hybrid Gravitational Search Algorithm

The Taillard [9] dataset was used during the test runs. The key features of the Taillard dataset is the following:

- Variety of problem sizes: The dataset includes examples for different numbers of machines (m) and workpieces (n) (e.g., 20×5, 50×10, 100×20), allowing testing of small and large scale problems.
- Independent, randomly generated processing times: The processing times are chosen so that the problems are not biased or trivially solvable.

- Wide acceptance: The benchmark is a quasi-standard in FSSP research and serves as a reference for testing almost all modern metaheuristic and exactness algorithms.
- The objective function of the problems is the makespan minimization

3.1. Particle Swarm Optimization test results

This section presents the results of Particle Swarm Optimization. First, the maximum, average, and minimum of the test runs are presented. Then, the section examines the test results in comparison to the benchmark algorithms.

Table 1. Fitness values of Particle Swarm Optimization: maximum, average and minimum values.

Instance	**PSO**		
	Max	Avg	Min
Ta001	1313	1301.8	1297
Ta002	1373	1368.4	1366
Ta003	1161	1153.2	1145
Ta004	1391	1380.6	1372
Ta005	1288	1284.6	1277
Ta006	1258	1248	1238
Ta007	1273	1262.4	1252
Ta008	1297	1281.6	1271
Ta009	1298	1288.2	1277
Ta010	1177	1170.6	1161
Ta011	1725	1713	1708
Ta012	1792	1785.4	1778
Ta013	1627	1616.2	1599
Ta014	1509	1493.8	1469
Ta015	1562	1552.6	1546
Ta016	1530	1509.6	1493
Ta017	1599	1586.8	1571
Ta018	1679	1667.6	1631
Ta019	1701	1693	1686
Ta020	1728	1711	1692

The Table 1 shows the performance of the Particle Swarm Optimization (PSO) algorithm on the Taillard dataset. The average fitness values of the solutions is close to the best value. This means, that PSO gives relatively stable results. However, differences can be observed between individual instances. For example, in the case of Ta018, where the difference between the Max and Min values is large (1679 and 1631 fitness values). In the case of Ta011, the difference is minimal. The algorithm achieved the lowest values for problems Ta003 (minimum fitness value: 1145) and

Ta010 (minimum fitness value: 1161). The worst result was on problem Ta012 (maximum fitness value: 1792).

Table 2. Comparison of test results obtained by Particle Swarm Optimization and competing algorithms.

Instance	PSO	HMM-PFA %	HGA %	IIGA %	DSOMA %	HGSA %
Ta001	1297	14.57	11.72	14.57	5.94	2.08
Ta002	1366	11.86	6.88	11.86	3.07	5.56
Ta003	1145	27.51	21.05	27.51	11.79	-4.10
Ta004	1372	15.74	10.86	15.74	5.54	7.07
Ta005	1277	13.47	9.87	13.47	5.01	1.10
Ta006	1238	19.63	15.51	19.63	10.10	12.36
Ta007	1252	18.45	16.69	18.45	10.30	3.75
Ta008	1271	16.60	12.75	16.60	8.50	1.65
Ta009	1277	15.04	9.48	15.04	7.52	2.27
Ta010	1161	18.60	14.04	18.60	10.51	6.20
Ta011	1708	19.67	14.46	17.74	-0.59	0.29
Ta012	1778	21.82	19.40	21.82	3.09	-3.37
Ta013	1599	21.33	19.57	21.33	4.82	-2.75
Ta014	1469	23.28	21.31	23.28	5.24	3.20
Ta015	1546	25.03	25.03	25.03	4.59	1.75
Ta016	1493	26.72	22.37	26.72	6.50	-2.41
Ta017	1571	24.95	23.74	24.95	3.25	3.25
Ta018	1631	26.12	22.99	26.12	6.13	7.23
Ta019	1686	17.02	13.17	17.02	3.62	-3.68
Ta020	1692	21.22	18.26	21.22	5.32	1.77

Table 2 compares the maximum of the test values of the Particle Swarm Optimization algorithm and the results of the algorithms published by the researchers. In some cases, the HGSA algorithm and in one case the DSOMA algorithm gave better results than the PSO algorithm. In the majority of cases (in the range of about 10-20 %) PSO gave better results than the other algorithms. In some cases (e.g. Ta001, Ta006, Ta018) HGSA gave better or similar results than PSO. In one case (Ta003) DSOMA achieved better results (-4.10 %) than PSO.

3.2. Tabu Search test results

In the following, the test results of the Tabu Search algorithm and their comparison with the results of algorithms published by researchers are presented.

The results of Tabu Search are presented in Table 3. The results show that the algorithm typically provides stable and reliable performance. In most cases, the maximum and average values are relatively close to each other. The method usually gives consistently good results. For example, for problem Ta006, the best fitness value is 1296, the average value is 1272.2, and the worst is 1233.

Table 4 compares the Tabu Search results with the published results. Only five cases was the proposed algorithm worse than the published results. In many cases,

Table 3. Fitness values of Tabu Search algorithm: maximum, average and minimum values.

Instance	**TS**		
	Max	Avg	Min
Ta001	1377	1356.6	1323
Ta002	1447	1405.6	1383
Ta003	1198	1166.2	1132
Ta004	1444	1405	1359
Ta005	1416	1333.2	1279
Ta006	1296	1272.2	1233
Ta007	1294	1275	1259
Ta008	1297	1269.2	1226
Ta009	1376	1321.6	1265
Ta010	1180	1162.4	1137
Ta011	1760	1723.4	1681
Ta012	1852	1784.6	1737
Ta013	1656	1613.8	1591
Ta014	1609	1518.6	1425
Ta015	1593	1553.6	1495
Ta016	1515	1487.8	1467
Ta017	1649	1611.2	1576
Ta018	1722	1679.8	1641
Ta019	1745	1701.8	1672
Ta020	1751	1720.2	1657

the TS algorithm gave more than 20% better results. The IIGA and HMM-PFA algorithms were similar or worse than TS. The DSOMA and HGSA algorithms gave significantly worse results. In most cases, TS gave results that were more than 20% better than the other algorithms, for example, for Ta003, IIGA is 28.98%. TS only performed worse in five cases: for Ta011 and Ta012, in the case of HGSA and DSOMA, and for Ta019 and Ta016, in the case of HGSA and DSOMA.

4. Conclusions and future research directions

The paper investigated the effectiveness of Particle Swarm Optimization and Tabu Search on the Flow Shop Scheduling Problem. The Taillard benchmark dataset was used for the problem. It can be said that TS was the most efficient algorithm. Most of the compared algorithms, such as HMM-PFA, HGA, IIGA, performed worse than the proposed algorithms. Particle Swarm Optimization (PSO) and Tabu Search (TS) algorithms give several future research directions, for example the hybridization of PSO and TS. Further research can also focus on combining them with other metaheuristic algorithms, such as Genetic Algorithm or Simulated

Table 4. Comparison of test results obtained by Tabu Search and competing algorithms.

Instance	TS	HMM-PFA %	HGA %	IIGA %	DSOMA %	HGSA %
Ta001	1323	12.32	9.52	12.32	3.85	0.08
Ta002	1383	10.48	5.57	10.48	1.81	4.27
Ta003	1132	28.98	22.44	28.98	13.07	-3.00
Ta004	1359	16.85	11.92	16.85	6.55	8.09
Ta005	1279	13.29	9.70	13.29	4.85	0.94
Ta006	1233	20.11	15.98	20.11	10.54	12.81
Ta007	1259	17.79	16.04	17.79	9.69	3.18
Ta008	1226	20.88	16.88	20.88	12.48	5.38
Ta009	1265	16.13	10.51	16.13	8.54	3.24
Ta010	1137	21.11	16.45	21.11	12.84	8.44
Ta011	1681	21.59	16.30	19.63	1.01	1.90
Ta012	1737	24.70	22.22	24.70	5.53	-1.09
Ta013	1591	21.94	20.18	21.94	5.34	-2.26
Ta014	1425	27.09	25.05	27.09	8.49	6.39
Ta015	1495	29.30	29.30	29.30	8.16	5.22
Ta016	1467	28.97	24.54	28.97	8.38	-0.68
Ta017	1576	24.56	23.35	24.56	2.92	2.92
Ta018	1641	25.35	22.24	25.35	5.48	6.58
Ta019	1672	18.00	14.11	18.00	4.49	-2.87
Ta020	1657	23.78	20.76	23.78	7.54	3.92

Annealing. Optimization of PSO and TS parameters can also be another future research direction.

References

[1] P. Agrawal, H. Abutarboush, T. Ganesh, A. Mohamed: *Metaheuristic algorithms on feature selection: A survey of one decade of research (2009-2019)*, IEEE Access 9 (2021), pp. 26766–26791, doi: 10.1109/ACCESS.2021.3056407.

[2] O. Arik: *Population-based Tabu search with evolutionary strategies for permutation flow shop scheduling problems under effects of position-dependent learning and linear deterioration*, Soft Computing 25.2 (2021), pp. 1501–1518, doi: 10.1007/s00500-020-05234-7.

[3] A. Gad: *Particle swarm optimization algorithm and its applications: a systematic review*, Archives of Computational Methods in Engineering 29.5 (2022), pp. 2531–2561, doi: 10.1007/s11831-021-09694-4.

[4] H. Lasi, P. Fettke, H. Kemper, T. Feld, M. Hoffmann: *Industry 4.0*, Business & Information Systems Engineering 6 (2014), pp. 239–242, doi: 10.1007/s12599-014-0334-4.

[5] Y. Li, C. Wang, L. Gao, Y. Song, X. Li: *An improved simulated annealing algorithm based on residual network for permutation flow shop scheduling*, Complex & Intelligent Systems 7 (2021), pp. 1173–1183, doi: 10.1007/s40747-020-00205-9.

[6] J. Neufeld, S. Schulz, U. Buscher: *A systematic review of multi-objective hybrid flow shop scheduling*, European Journal of Operational Research 309.1 (2023), pp. 1–23, doi: 10.1016/j.ejor.2022.08.009.

[7] M. Parente, G. Figueira, P. Amorim, A. Marques: *Production scheduling in the context of Industry 4.0: review and trends*, International Journal of Production Research 58.17 (2020), pp. 5401–5431, doi: 10.1080/00207543.2020.1718794.

[8] C. Qu, Y. Fu, Z. Yi, J. Tan: *Solutions to no-wait flow shop scheduling problem using the flower pollination algorithm based on the hormone modulation mechanism*, Complexity (2018), doi: 10.1155/2018/1973604.

[9] E. Taillard: *Benchmarks for basic scheduling problems*, European Journal of Operational Research 64.2 (1993), pp. 278–285, doi: 10.1016/0377-2217(93)90182-M.

[10] K.-P. Wang, L. Huang, C.-G. Zhou, W. Pang: *Particle swarm optimization for traveling salesman problem*, in: Proceedings of the 2003 international conference on machine learning and cybernetics (IEEE cat. no. 03ex693), vol. 3, IEEE, 2003, pp. 1583–1585.

[11] H. Wei, S. Li, H. Jiang, J. Hu: *Hybrid genetic simulated annealing algorithm for improved flow shop scheduling with makespan criterion*, Applied Sciences 8.12 (2018), p. 2621, doi: 10.3390/app8122621.

[12] J. Xie, X. Li, L. Gao, L. Gui: *A hybrid genetic tabu search algorithm for distributed flexible job shop scheduling problems*, Journal of Manufacturing Systems 71 (2023), pp. 82–94, doi: 10.1016/j.jmsy.2023.09.002.

[13] W. Xu, L. He, G. Zhu: *Many-objective flow shop scheduling optimisation with genetic algorithm based on fuzzy sets*, International Journal of Production Research 59.3 (2021), pp. 702–726, doi: 10.1080/00207543.2019.1705418.

[14] W. Zhang, W. Hou, C. Li, W. Yang, M. Gen: *Multidirection update-based multiobjective particle swarm optimization for mixed no-idle flow-shop scheduling problem*, Complex System Modeling and Simulation 1.3 (2021), pp. 176–197, doi: 10.23919/CSMS.2021.0017.