Production Systems and Information Engineering
Volume 12 (3), pp. 1-10
doi: https://doi.org/10.32968/psaie.2024.3.1

Efficiency analysis of the Ant Colony System algorithm in benchmark data

ANITA AGÁRDI
University of Miskolc, Hungary
Institute of Information Technology
agardianita@iit.uni-miskolc.hu

Abstract.
The paper analyzes the effectiveness of the Ant Colony System (ACS) on a production scheduling dataset. The Ant Colony System (ACS) algorithm is a member of the Ant Colony Optimization (ACO) algorithm family, a discrete optimization algorithm. The article presents the ACS algorithm, the production scheduling task and the test results, the analysis of the benchmark dataset, and the search space of the algorithm.

Keywords: Ant Colony System, production scheduling, fitness landscape analysis

1. Introduction

Over the years, optimization algorithms have become more and more important due to industrial needs. The goal is to develop a more cost-effective solution proposal for complex systems. Over the years, many optimization algorithms have been published. In this paper, we investigate the effectiveness of the Ant System [1], a variant of Ant Colony Optimization (ACO), on a production scheduling problem, which is the Flow Shop Scheduling (FSS) Problem [2].
First, with the help of Google Scholar, I conducted a literature search to see how many articles have been published over the years about the Ant Colony System. I conducted the literature research from 2010 to 2023. The Ant Colony System algorithm was first published in 1996 in the article [1] by Marco Dorigo and Luca Maria Gambardella.

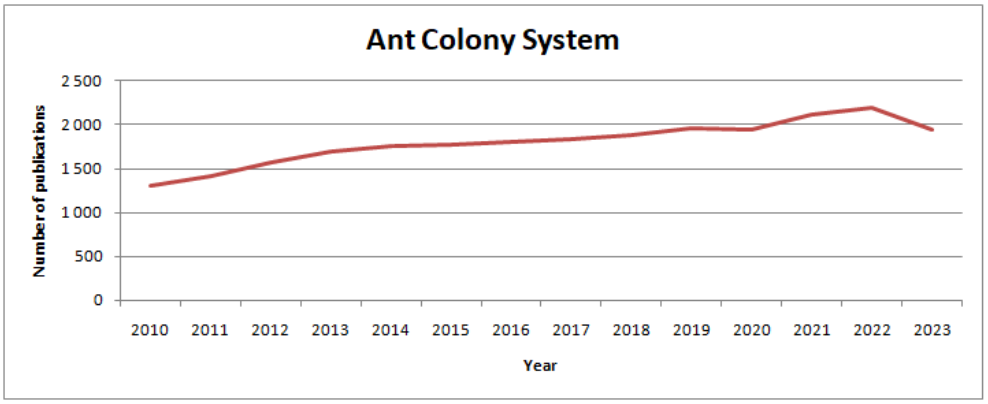

Figure 1. Ant Colony System
keyword result in Google Scholar

With the Ant Colony System algorithm, we can see that over the years researchers have published more and more articles, although slightly fewer articles were published in 2023 than in previous years. 1310 articles were published in 2010, 1780 articles in 2015, 1940 articles in 2020, and 1950 in 2023.
In the following parts of the article, the Ant Colony System algorithm, the Flow Shop Scheduling Problem, and the test results are presented. The last chapter contains the conclusion and future research direction.

2. Ant Colony System and Flow Shop Scheduling Problem

2.1. Ant Colony System [1]

This algorithm is a type of Ant Colony Optimization (ACO) algorithm. The ACO algorithm is based on the behavior of ants. The ants' search for food is modeled by the algorithm family. The ants will choose the road section that is short and has high pheromone content. Individual ants deposit pheromones during their journey. The pheromone, on the other hand, evaporates over time. These processes are modeled by ACO algorithms. The steps of the algorithm are as follows:

1. Initialization of input parameters and initialization of ants (e.g. with random path)
2. Local search
3. Re-updating the pheromone contents
4. Repeat steps 2-3 until the stop condition is met

The Ant Colony System constructs its route using the formula below:

$$p_{ij}^{k} = \frac{[\tau_{ij}(t)]^{\alpha} * [\eta_{ij}]^{\beta}}{\sum_{l \in N_i^k} [\tau_{il}(t)]^{\alpha} * [\eta_{il}]^{\beta}} \quad if\ j \in N_i^k$$

$\eta_{ij} = \frac{1}{d_{ij}}$: the reciprocal of the distance between the two routes
$\tau_{ij}(t)$: pheromone content
(i, j): edge pair
α and β: the effects of the pheromone and the distance are determined by the parameters
N_i^k: the nodes that the ant has not yet visited

Ant Colony System has both global and local pheromone updates.
Global Pheromone Upgrade Formula:

$$\tau_{ij}(t+1) = (1-\rho) * \tau_{ij}(t) + \rho * \Delta\tau_{ij}^{gb}(t)$$

where: $\Delta\tau_{ij}^{gb}(t) = \frac{1}{L^{gb}}$

Local pheromone update:

$$\tau_{ij} = (1-\xi) * \tau_{ij} + \xi * {\tau_{ij}}^{0}$$

where:
$0 < \xi < 1$:
${\tau_{ij}}^{0}$: initial pheromone value

The Ant Colony System algorithm has been used for many discrete optimization tasks over the years; I would like to highlight a few of them:

- Traveling Salesman Problem [1]
- Vehicle Routing Problem [3]
- unit commitment problem [4]

- Multiobjective cloud workflow scheduling [5]
- Coverage path planning [6]
- airline crew rostering problem [7]

2.2. Flow Shop Scheduling Problem [2]

The Flow Shop Scheduling Problem is a production scheduling problem where a given number of tasks must be performed on a given number of machines. During the task, the goal is to minimize the makespan. Each job must be performed exactly once on each machine. Once the machines have started a job, they must finish it before they start the next job. There is also a production time until the given machine performs the given work, which also includes the refitting time. Each machine must do each job exactly once.
The Flow Shop Scheduling problem has been solved over the years with several algorithms; I would like to highlight a few of them:

- Genetic algorithm [8]
- Tabu Search [8]
- whale optimization algorithm [9]
- Iterated greedy algorithm [10]
- reinforcement learning algorithm [11]
- neural network [11]
- water wave optimization algorithm [12]
- Improved Q-learning algorithm [13]
- mutant firefly algorithm [14]

Several versions of Flow Shop Scheduling have evolved over the years, adapting to the real industrial needs of the time. I would like to highlight a few of them:

- no-wait flow shop scheduling problem [15]
- Flow shop scheduling problems with assembly operations [16]
- flow shop scheduling problems with setup energy consumptions [17]
- total tardiness parallel blocking flow shop scheduling problem [18]
- flow shop scheduling problem with limited human resource constraints [19]
- flow shop scheduling problem with sequence-independent setup time [20]
- flow shop scheduling problem with missing operations [21]
- Multi-machine flow shop scheduling problems with rejection [22]
- Flow shop scheduling with jobs arriving at different times [23]
- energy-efficient scheduling of no-wait flow-shop problem [24]
- no-idle flow shop scheduling problem [25]
- Flow shop scheduling with blocking [26]

3. Test results

The table below shows the running results of the Ant Colony System compared to the other algorithms. I used the Taillard benchmark dataset [27] for the tests. The table consists of three parts, the first part contains the elements HMM-PFA, HGA, IIGA, DSOMA, and HGSA. These algorithms are not implemented in the frame of this research, but published by others. Here you can find results for each data line.
The second part contains only MA and IG-RIS because these algorithms had data from Ta031 to Ta080. And the third table contains the IWO algorithm because here there were data for the following data series: Ta001, Ta011, Ta021, Ta031, Ta041, Ta051, Ta061, Ta071.

Table 2. Test result comparison (HMM-PFA, HGA, IIGA, DSOMA, HGSA)

		Relative performance				
Instance	**ACS**	**HMM-PFA [28] %**	**HGA [29] %**	**IIGA [29] %**	**DSOMA [29] %**	**HGSA [29] %**
Ta001	1297	114.57	113.64	114.57	105.94	102.08
Ta002	1368	111.70	109.05	111.7	102.92	105.41
Ta003	1138	128.30	112.78	128.3	112.48	96.49
Ta004	1362	116.59	115.68	116.59	106.31	107.86
Ta005	1281	113.11	120.28	113.11	104.68	100.78
Ta006	1234	120.02	119.57	120.02	110.45	112.72
Ta007	1259	117.79	121.8	117.79	109.69	103.18
Ta008	1261	117.53	126.59	117.53	109.36	102.46
Ta009	1282	114.59	121.88	114.59	107.1	101.87
Ta010	1174	117.29	122.34	117.29	109.28	105.03
Ta011	1690	120.95	120.84	118.99	100.47	101.36
Ta012	1765	122.72	112.9	122.72	103.85	97.34
Ta013	1599	121.33	118.68	121.33	104.82	97.25
Ta014	1463	123.79	120.93	123.79	105.67	103.62
Ta015	1527	126.59	123.67	126.59	105.89	103.01
Ta016	1499	126.22	119.8	126.22	106.07	97.2
Ta017	1589	123.54	126.47	123.54	102.08	102.08
Ta018	1660	123.92	120.93	123.92	104.28	105.36
Ta019	1690	116.75	123.8	116.75	103.37	96.09
Ta020	1686	121.65	124.22	121.65	105.69	102.14
Ta021	2408	123.46	119.3	123.46	101.16	96.8
Ta022	2248	126.87	124.2	114.86	99.38	101.42
Ta023	2439	123.53	125.12	123.53	101.64	101.68
Ta024	2346	127.92	112.97	127.92	100.09	100.68
Ta025	2442	122.97	116.42	122.97	99.71	102.66
Ta026	2349	127.63	117.55	127.2	101.45	101.11
Ta027	2391	127.65	115.44	127.65	99.96	97.91
Ta028	2316	122.58	113.1	122.58	100.52	98.4
Ta029	2393	125.74	113.6	125.74	98.75	100.71
Ta030	2333	127.69	113.44	127.69	99.57	102.91
Ta031	2768	114.16	116.7	114.2	109.57	98.66
Ta032	2953	116.22	115.22	116.22	103.12	99.36
Ta033	2707	118.58	116.03	118.62	112.15	97.45
Ta034	2849	117.16	125.62	117.2	105.69	97.75
Ta035	2931	114.50	128.74	114.5	106.72	97.71
Ta036	2926	114.35	126.88	114.39	108.2	99.35
Ta037	2806	115.15	132.82	115.15	107.66	98.5
Ta038	2779	116.41	128.57	116.41	110.22	97.37
Ta039	2655	115.63	128.36	115.71	109.53	98.31
Ta040	2845	116.59	128.33	116.59	109.67	97.86
Ta041	3384	126.30	129.17	126.3	107.51	94.5
Ta042	3215	129.92	128.26	129.92	109.21	93.93
Ta043	3218	127.38	123.53	127.38	108.51	94.93
Ta044	3373	130.42	142.81	130.42	108.86	92.62
Ta045	3357	128.75	135.41	128.75	108.22	93.21
Ta046	3336	128.57	139.68	128.57	108.54	98.71
Ta047	3410	129.62	138.37	129.62	108.62	94.78
Ta048	3332	129.59	140.26	129.59	107.2	101.74
Ta049	3245	128.04	140.15	128.04	109.12	99.75

Ta050	3426	125.01	139.2	125.01	105.78	94.89
Ta051	4298	142.60	138.81	142.6	104.96	95.51
Ta052	4225	135.50	136.84	135.5	101.49	94.49
Ta053	4186	140.04	137.29	140.04	102.46	93.17
Ta054	4178	138.54	116.62	138.54	104.79	93.85
Ta055	4200	140.14	117.26	140.14	101.69	95.71
Ta056	4192	139.86	115.26	139.86	100.24	94.73
Ta057	4265	139.79	118.1	139.79	101.17	95.97
Ta058	4264	138.98	116.09	138.98	101.45	95.92
Ta059	4267	137.71	117.7	137.71	101.45	96.25
Ta060	4323	137.80	116.04	137.82	102.29	95.14
Ta061	5567	114.26	117.81	114.91	110.49	99.44
Ta062	5418	114.65	114.91	115.06	111.92	97.86
Ta063	5334	114.44	117.05	114.75	112.54	97.88
Ta064	5148	116.53	129.2	117.06	112.39	97.98
Ta065	5387	114.70	135.75	115.09	111.77	99.46
Ta066	5248	115.40	132.08	115.74	111.83	99.03
Ta067	5392	115.37	130.8	116.36	111.35	100.41
Ta068	5284	115.61	131.93	116.01	112.11	97.09
Ta069	5607	113.34	134.42	113.61	109.76	98.91
Ta070	5503	115.66	131.46	115.95	112.41	99.58
Ta071	6281	128.24	121.07	128.59	112.12	94.95
Ta072	5883	133.49	130.28	133.95	115.81	95.12
Ta073	6100	131.41	131.29	131.61	113.82	95.02
Ta074	6366	130.82	113.64	131.13	113.07	93.12
Ta075	6057	131.02	109.05	129.75	112.51	94.9
Ta076	5820	133.56	112.78	134.04	114.86	93.57
Ta077	6021	130.31	115.68	130.64	113.39	94.32
Ta078	6095	129.29	120.28	129.83	112.78	93.9
Ta079	6314	128.78	119.57	129.25	96.48	93.98
Ta080	6235	129.78	121.8	130.14	112.11	96.2

Table 3. Test result comparison (MA, IG-RIS)

		Relative performance	
Instance	**ACS**	**MA [29] %**	**IG-RIS [29] %**
Ta031	2768	108.38	108.45
Ta032	2953	108.33	108.4
Ta033	2707	111.23	111.23
Ta034	2849	109.79	109.79
Ta035	2931	107.88	108.02
Ta036	2926	108.2	108.3
Ta037	2806	107.38	107.38
Ta038	2779	110.36	110.58
Ta039	2655	109.53	109.53
Ta040	2845	109.35	109.67
Ta041	3384	107.51	107.51
Ta042	3215	108.43	109.08
Ta043	3218	108.23	108.39
Ta044	3373	108.39	108.39
Ta045	3357	108.1	108.1
Ta046	3336	107.79	108.54
Ta047	3410	108.27	108.39
Ta048	3332	106.9	107.2
Ta049	3245	108.69	108.84
Ta050	3426	105.72	105.78

Ta051	4298	104.21	104.7
Ta052	4225	101.21	101.21
Ta053	4186	101.79	102.46
Ta054	4178	104.5	104.76
Ta055	4200	101.45	101.62
Ta056	4192	102.1	102.1
Ta057	4265	100.91	101.01
Ta058	4264	101.24	101.45
Ta059	4267	101.12	101.15
Ta060	4323	102.08	102.43
Ta061	5567	110.35	110.49
Ta062	5418	111.15	111.15
Ta063	5334	111.12	111.12
Ta064	5148	111.81	112.12
Ta065	5387	110.58	110.64
Ta066	5248	110.75	111.51
Ta067	5392	111.07	111.35
Ta068	5284	110.83	111.94
Ta069	5607	108.19	109.2
Ta070	5503	111.61	111.92
Ta071	6281	111.7	112.12
Ta072	5883	114.57	115.43
Ta073	6100	112.75	113.7
Ta074	6366	111.78	112.9
Ta075	6057	112.43	112.43
Ta076	5820	113.64	114.54
Ta077	6021	112.66	112.95
Ta078	6095	111.4	112.78
Ta079	6314	110.56	111.74
Ta080	6235	109.29	111.71

Table 4. Test result comparison
(IWO)

		Relative performance
Instance	**ACS**	**IWO [30]%**
Ta001	1297	107.09
Ta011	1690	130.59
Ta021	2408	133.97
Ta031	2768	109.1
Ta041	3384	102.39
Ta051	4298	127.38
Ta061	5567	104.89
Ta071	6281	108.5

The table shows the results of the Ant Colony Optimization algorithm and the results compared to other algorithms in percentage.
I ran the Ant Colony Optimization (ACO) algorithm up to the Ta080 dataset. The table contains the comparisons to the algorithms. I illustrate in a table how many cases the ACO algorithm is better than the other algorithms.

Table 5. Test result comparison

Algorithm	**Number of data rows (on which the comparisons were made)**	**Number of better results**
HMM-PFA	80	80

MA	50	50
IG-RIS	50	50
HGA	80	80
IIGA	80	80
DSOMA	80	74
HGSA	80	24
IWO	8	8

It can be seen from the tables that in the case of HMM-PFA, MA, IG-RIS, HGA, and IIGA, the ACO algorithm always gave better results. In the case of DSOMA, it did not give better results only a few times. It gave better results than the HGSA and IWO algorithms in only a few cases, but the efficiency of ACO was not far behind.

The results of the Ant Colony System (ACS) algorithm search space are shown in the table below. The solutions given by the iterations of the Ant Colony System algorithm were compared to each other. The following metrics were used here [31]:

- Lower and upper limits of fitness values
- The lower and upper limits of the averages of the fitness Hamming and Basic Swap Sequence distances taken from each other
- Cost Density, which shows how different the solutions given by the iterations are, i.e. how many solutions have the same fitness value.

The essence of fitness landscape analysis [31] is to determine, which algorithms are outperformed others, and why. It is an analytical analysis, based on different metrics. In this paper, this measurement uses the best results given by the iterations of the algorithm. The general purpose of the fitness landscape can be multiple: deciding the difficulty of the problem itself, examining the effectiveness of each algorithm, examining the operators used.

Table 6. Fitness landscape analysis

Ant Colony System (ACS)											
		Ta001		**Ta002**		**Ta003**		**Ta004**		**Ta005**	
	Distance	**LB**	**UB**	**LB**	**UB**	**LB**	**UB**	**LB**	**UB**	**LB**	**UB**
Fitness values		1328	1347	1376	1402	1180	1257	1410	1480	1302	1360
Average of fitness distances	**Fitness**	2.02	47.98	2.55	12.45	6.55	43.03	5.24	64.76	3.39	69.61
Average of Hamming distances	**Hamming**	10.5	18.65	3.4	16.6	13.26	18.34	9.39	19.31	3.81	19.8
Average of basic swap sequence distances	**BSS**	9.03	17.03	2.72	13.28	7.09	15.93	7.83	16.81	3.59	17.21
Fitness distances of the best solution	**Fitness**	2.83	15.69	2.71	23.29	6.55	43.03	14.94	40.22	3.39	69.61
Hamming distances of the best solution	**Hamming**	1.62	17.82	4.94	18.66	13.26	18.34	2.46	19.49	3.81	19.8
Basic swap sequence distances of the best solution	**BSS**	1.43	14.85	4.47	16.89	10.9	16.12	2.26	17.0	3.59	17.21
Cost density		1.0	92.0	17.0	84.0	1.0	54.0	1.0	88.0	1.0	69.0

The above table shows the fitness landscape results of the Ant Colony System algorithm. The Ta001 fitness values were between 1347 and 1328 during the iterations, while the results for Ta002 were between 1257 and 1180, the values for Ta003 were between 1257 and 1180, and the fitness values for Ta004 were between 1480 and 1410. Based on this, it can be said that improvements were during each iteration, but the best result of the first iteration was a relatively good result. This was also shown by the averages of the fitness values. Hamming and Basic Swap

Sequence distances are quite large. The upper limits of Cost density are quite large, which indicates that after a few iterations, there is no improvement. For Ta001, this value was 92, and for Ta002, 84. For Ta003, there were also 54 equal solutions, while for Ta004, there were 88 equal solutions, and for Ta005, there were also 69 equal solutions.

4. Conclusions and future work

The Ant Colony System algorithm and its solution to a common production scheduling task, the Flow Shop Scheduling (FSS) problem, were presented in the article. The article contains a solution to a benchmark dataset. The article compares the results of the Ant Colony System with the following algorithms: HMM-PFA, MA, IG-RIS, HGA, IIGA, DSOMA, HGSA, and IWO. In most of the test results, ACS gave better results than the algorithms. Only the DSOMA and HGSA algorithms had results that were better than ACS. In the case of DSOMA, there were only 6 such cases, in the case of HGSA there were more, here there were 56 such cases, but ACS was only a few percent behind. The article also presented the analysis of the ACS fitness landscape, where the solutions given by the iterations were analyzed. The analyses were performed using the following techniques: lower and upper limits of the fitness values, also and upper limits of the average Hamming, fitness, and basic swap sequence distances.
Future research area is the analysis of other algorithms of the Ant Colony Optimization family for the Flow Shop Scheduling problem.

Acknowledgment.
„Supported by the ÚNKP-23-4-II. New National Excellence Program of the Ministry for Culture and Innovation from the source of the National Research, Development and Innovation Fund.”

References

[1] Dorigo, M., & Gambardella, L. M. (1997). Ant colony system: a cooperative learning approach to the traveling salesman problem. IEEE Transactions on evolutionary computation, 1(1), 53-66. https://doi.org/10.1109/4235.585892

[2] Kulcsar, G., & Erdélyi, Ferenc (2006). Modeling and solving of the extended flexible flow shop scheduling problem. Production Systems and Information Engineering, 3, 121-139.

[3] Montemanni, R., Gambardella, L. M., Rizzoli, A. E., & Donati, A. V. (2005). Ant colony system for a dynamic vehicle routing problem. Journal of combinatorial optimization, 10, 327-343. https://doi.org/10.1007/s10878-005-4922-6

[4] Simon, S. P., Padhy, N. P., & Anand, R. S. (2006). An ant colony system approach for unit commitment problem. International Journal of Electrical Power & Energy Systems, 28(5), 315-323. https://doi.org/10.1016/j.ijepes.2005.12.004

[5] Chen, Z. G., Zhan, Z. H., Lin, Y., Gong, Y. J., Gu, T. L., Zhao, F., ... & Zhang, J. (2018). Multiobjective cloud workflow scheduling: A multiple populations ant colony system approach. IEEE transactions on cybernetics, 49(8), 2912-2926. https://doi.org/10.1109/TCYB.2018.2832640

[6] Chen, J., Ling, F., Zhang, Y., You, T., Liu, Y., & Du, X. (2022). Coverage path planning of heterogeneous unmanned aerial vehicles based on ant colony system. Swarm and Evolutionary Computation, 69, 101005.

https://doi.org/10.1016/j.swevo.2021.101005

[7] Zhou, S. Z., Zhan, Z. H., Chen, Z. G., Kwong, S., & Zhang, J. (2020). A multi-objective ant colony system algorithm for airline crew rostering problem with fairness and satisfaction. IEEE Transactions on Intelligent Transportation Systems, 22(11), 6784-6798. https://doi.org/10.1109/TITS.2020.2994779

[8] Umam, M. S., Mustafid, M., & Suryono, S. (2022). A hybrid genetic algorithm and tabu search for minimizing makespan in flow shop scheduling problem. Journal of King Saud University-Computer and Information Sciences, 34(9), 7459-7467. https://doi.org/10.1016/j.jksuci.2021.08.025

[9] Abdel-Basset, M., Manogaran, G., El-Shahat, D., & Mirjalili, S. (2018). A hybrid whale optimization algorithm based on local search strategy for the permutation flow shop scheduling problem. Future generation computer systems, 85(1), 129-145. https://doi.org/10.1016/j.future.2018.03.020

[10] Zhao, Z., Zhou, M., & Liu, S. (2021). Iterated greedy algorithms for flow-shop scheduling problems: A tutorial. IEEE Transactions on Automation Science and Engineering, 19(3), 1941-1959. https://doi.org/10.1109/TASE.2021.3062994

[11] Ren, J., Ye, C., & Yang, F. (2021). Solving flow-shop scheduling problem with a reinforcement learning algorithm that generalizes the value function with neural network. Alexandria Engineering Journal, 60(3), 2787-2800. https://doi.org/10.1016/j.aej.2021.01.030

[12] Zhao, F., Liu, H., Zhang, Y., Ma, W., & Zhang, C. (2018). A discrete water wave optimization algorithm for no-wait flow shop scheduling problem. Expert Systems with Applications, 91, 347-363. https://doi.org/10.1016/j.eswa.2017.09.028

[13] He, Z., Wang, K., Li, H., Song, H., Lin, Z., Gao, K., & Sadollah, A. (2022). Improved Q-learning algorithm for solving permutation flow shop scheduling problems. IET Collaborative Intelligent Manufacturing, 4(1), 35-44. https://doi.org/10.1049/cim2.12042

[14] Fan, B., Yang, W., & Zhang, Z. (2019). Solving the two-stage hybrid flow shop scheduling problem based on mutant firefly algorithm. Journal of Ambient Intelligence and Humanized Computing, 10, 979-990. https://doi.org/10.1007/s12652-018-0903-3

[15] Engin, O., & Güçlü, A. (2018). A new hybrid ant colony optimization algorithm for solving the no-wait flow shop scheduling problems. Applied Soft Computing, 72, 166-176. https://doi.org/10.1016/j.asoc.2018.08.002

[16] Komaki, G. M., Sheikh, S., & Malakooti, B. (2019). Flow shop scheduling problems with assembly operations: a review and new trends. International Journal of Production Research, 57(10), 2926-2955. https://doi.org/10.1080/00207543.2018.1550269

[17] Li, J. Q., Sang, H. Y., Han, Y. Y., Wang, C. G., & Gao, K. Z. (2018). Efficient multi-objective optimization algorithm for hybrid flow shop scheduling problems with setup energy consumptions. Journal of Cleaner Production, 181, 584-598. https://doi.org/10.1016/j.jclepro.2018.02.004

[18] Ribas, I., Companys, R., & Tort-Martorell, X. (2019). An iterated greedy algorithm for solving the total tardiness parallel blocking flow shop scheduling problem. Expert Systems with Applications, 121, 347-361. https://doi.org/10.1016/j.eswa.2018.12.039

[19] Costa, A., Fernandez-Viagas, V., & Framiñan, J. M. (2020). Solving the hybrid flow shop scheduling problem with limited human resource constraint. Computers & Industrial Engineering, 146, 106545. https://doi.org/10.1016/j.cie.2020.106545

[20] Belabid, J., Aqil, S., & Allali, K. (2020). Solving permutation flow shop scheduling problem with sequence-independent setup time. Journal of Applied Mathematics, 2020, 1-11. https://doi.org/10.1155/2020/7132469

[21] Rossit, D. A., Toncovich, A. A., Rossit, D. G., & Nesmachnow, S. (2020). Solving a flow shop scheduling problem with missing operations in an Industry 4.0 production environment.

[22] Dabiri, M., Darestani, S. A., & Naderi, B. (2019). Multi-machine flow shop

scheduling problems with rejection using genetic algorithm. International Journal of Services and Operations Management, 32(2), 158-172. https://doi.org/10.1504/IJSOM.2019.097527

[23] Li, G., Li, N., Sambandam, N., Sethi, S. P., & Zhang, F. (2018). Flow shop scheduling with jobs arriving at different times. International Journal of Production Economics, 206, 250-260. https://doi.org/10.1016/j.ijpe.2018.10.010

[24] Zhao, F., He, X., & Wang, L. (2020). A two-stage cooperative evolutionary algorithm with problem-specific knowledge for energy-efficient scheduling of no-wait flow-shop problem. IEEE transactions on cybernetics, 51(11), 5291-5303. https://doi.org/10.1109/TCYB.2020.3025662

[25] Shao, W., Pi, D., & Shao, Z. (2018). Local search methods for a distributed assembly no-idle flow shop scheduling problem. IEEE Systems Journal, 13(2), 1945-1956. https://doi.org/10.1109/JSYST.2018.2825337

[26] Doush, I. A., Al-Betar, M. A., Awadallah, M. A., Santos, E., Hammouri, A. I., Mafarjeh, M., & AlMeraj, Z. (2019). Flow shop scheduling with blocking using modified harmony search algorithm with neighboring heuristics methods. Applied Soft Computing, 85, 105861. https://doi.org/10.1016/j.asoc.2019.105861

[27] E. Taillard, "Benchmarks for basic scheduling problems", EJOR 64(2):278-285, 1993. https://doi.org/10.1016/0377-2217(93)90182-M

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

[29] Wei, H., Li, S., Jiang, H., Hu, J., & Hu, J. (2018). Hybrid genetic simulated annealing algorithm for improved flow shop scheduling with makespan criterion. Applied Sciences, 8(12), 2621. https://doi.org/10.3390/app8122621

[30] Zhou, Y., Chen, H., & Zhou, G. (2014). Invasive weed optimization algorithm for optimization no-idle flow shop scheduling problem. Neurocomputing, 137, 285-292. https://doi.org/10.1016/j.neucom.2013.05.063

[31] Agárdi, A., Kovács, L., & Bányai, T. (2021). The fitness landscape analysis of the ant colony system algorithm in solving a vehicle routing problem. ACADEMIC JOURNAL OF MANUFACTURING ENGINEERING, 19(2).