Multidiszciplináris Tudományok, 14. kötet, 4. szám (2024) pp. 217–228. **https://doi.org/10.35925/j.multi.2024.4.18**

SOLVING A PRODUCTION SCHEDULING PROBLEM WITH GENETIC ALGORITHM

Anita Agárdi
senior lecturer, University of Miskolc, Institute of Informatics
3515 Miskolc-Egyetemváros, e-mail: agardianita@iit.uni-miskolc.hu

Abstract
This article investigates a production scheduling task, the Flow Shop Scheduling (FSS) problem. The article solves the FSS task, during which a given number of tasks must be executed on a given number of machines, with a Genetic Algorithm (GA). The Genetic Algorithm is a population-based metaheuristic algorithm that maintains a population of solutions. It performs operations such as mutation and crossover on the current solutions until the stopping condition is not met. The article presents the effectiveness of the Genetic Algorithm on a benchmark data set, compared with six heuristic algorithms. The running results show that the Genetic Algorithm gave the best results in most of the test results.

Keywords: *Flow Shop Scheduling Problem, Genetic Algorithm*

1. Introduction

The article investigates a production scheduling problem, the Flow Shop Scheduling Problem (FSS). The article uses the Genetic Algorithm for the problem. The Flow Shop Scheduling Problem (FSS) (Kulcsár et al., 2007) is a production scheduling problem in which n tasks must be performed by m machines. Every task must be done by every machine. If a task is started by a machine, it must be finished before starting another task. The goal is to minimize the production time. Several versions of the Flow Shop Scheduling (FSS) task have developed over the years, which will be presented below.

- Permutation Flow Shop (Framinan et al., 2004): all jobs must pass through the same operations (machines), but the order of operations (machines) may vary.
- Linear Flow Shop (de Jong et al., 2017): all jobs must pass through the operations (machines) in the same order.
- Open Shop (Anand et al., 2015): jobs can be executed on any machine in any order.
- Job Shop (Mellor, 1966): the machines can perform the jobs in different sequences.
- Flexible Manufacturing System (FMS) (Yadav et al., 2018): enables the automatic reconfiguration of machines and tools during the production of different products. It enables adaptation to changing product types.
- Hybrid Flow Shop (Ruiz et al., 2010): it combines the characteristics of a flow shop and a job shop.
- Flow Shop with Limited Buffers (Jiang et al., 2019): jobs have only a limited buffer available between operations. Due to the limited size of the buffer, jobs must be efficiently scheduled between machines.
- Flow Shop with Parallel Machines (Nowicki et al., 1998): the jobs can be performed on several machines.

- Flow Shop with Sequence Dependent Setup Times (Javadian et al., 2010): setup times between machines are variable and depend on the order between jobs.
- Dynamic Flow Shop (Oukil et al., 2021): the availability of jobs and the operational readiness of machines may change over time; real-time decisions must also be taken into account.
- Flow Shop with Blocking (Wang et al., 2010): jobs must pass through machines where the next operation is not yet available (blocked).
- Flow Shop with No-Wait Constraint (Chen et al., 2020): jobs are not allowed to wait between operations.
- Flow Shop with Batching (Mirsanei et al., 2009): jobs can be organized into groups (batching), and these groups are done by the machines at the same time.
- Flow Shop with Preemption (Huo et al., 2016): execution of operations can be interrupted in the meantime and jobs can be transferred to another machine.
- Flow Shop with Time Windows (Chamnanlor, 2013): each operation has a time window within the jobs must be completed.
- Flow Shop with Energy Constraints (Jiang et al., 2019): machines has energy constraints and scheduling must consider the energy efficiency.
- Flow Shop with Multi-objective Optimization (Neufeld et al., 2023): several objective functions must be optimized for this task, e.g. makespan, costs.
- Flow Shop with Uncertain Processing Times (Kouvelis et al., 2000): operation times are uncertain.
- Flow Shop with Learning Effects (Azzouz et al., 2018): machines perform certain operations faster and more efficiently over time.
- Flow Shop with Machine Availability Constraints (Kis et al., 2006): the availability of machines may change from time to time.
- Flow Shop with Resource Flexibility (Emmons et al., 2013): production resources can be exchanged or used flexibly.
- Flow Shop with Multi-stage Batching (Li et al., 2015): jobs are grouped in different stages.
- Flow Shop with Weighted Tardiness (Xiao et al., 2012): in case of delay, the amount of the penalty varies.

The Genetic Algorithm is a population-based algorithm that maintains a population of solutions. Over the years, many related articles have been published. First, the article presents the results of Google Scholar, which can be seen in the figure below. The figure contains articles published from 2010 to 2023.

The articles related to the Genetic Algorithm have increased over the years, except for 2019-2022, in this period a small decrease can be observed, but in 2022-2023 there is already an increase.

While 54,700 articles were published in 2010, 76,600 articles were published in 2015, 92,100 articles in 2020, and 79,500 publications in 2023.

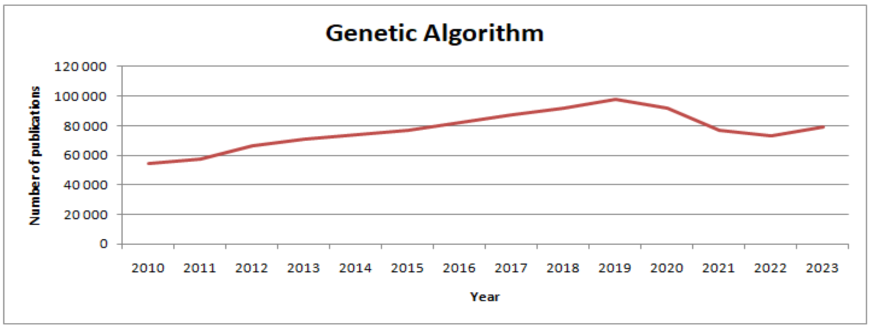

Figure 1. *Genetic Algorithm keyword result based on the Google Scholar*

The Genetic Algorithm has already been applied to many problems; I would like to highlight some of them below.

In the field of production scheduling, the Genetic Algorithm was applied to the following tasks: Manufacturing Process Scheduling (Burduk et al., 2019), Production Line Scheduling (Gubán et al., 2012), Flexibility in Manufacturing Processes (Jawahar et al., 1998), Timed Production Tasks (Fang, 1994), Resource Optimization (Hegazy et al., 2003), Balancing Local Resources (Dasgupta et al., 2013).

In the case of the vehicle route optimization task, a few tasks have already been highlighted below: Vehicle Route Planning (Pellazar, 1994), Fleet Distribution (Chakroborty et al., 2001), Transportation of Subunits (Chouhan et al., 2019), Flexible Transportation (Kubota et al., 1997), Combined Transportation (Oudani et al., 2014), Improving Fuel Efficiency (Xu et al., 2020), Transportation of Refrigerated or Sensitive Cargo (Zhang et al., 2019).

During the planning and optimization of neural networks, the algorithm can be used in the following tasks: Hyperparameter Tuning (Aszemi et al., 2019), Optimizing Layer Structure (Han et al., 1996), Weights and Biases Optimization (Rojas et al., 2022), Topological Optimization (Bevilacqua et al., 2006), Adaptive Learning Rate (Takahashi et al., 1993).

Genetic Algorithms can also be applied in the field of logistics, highlighting a few examples: Vehicle Route Planning (Pellazar, 1994), Warehouse Resource Optimization (Kordos et al., 2020), Inventory Strategies (Radhakrishnan et al., 2009), Transportation and Delivery Timing (Ongcunaruk et al., 2021), Transportation of Refrigerated and Sensitive Products (East et al., 2008), Flexible Transportation (Gupta et al., 2006), Network Logistics (Ko et al., 2007), Order Process Optimization (Bo et al., 2006).

In addition, many other areas could be mentioned, such as finance, bioinformatics, and combinatorial optimization problems.

The rest of the article is organized as follows: section 2 describes the genetic algorithm, and section 3 describes the results of the test run. Section 4 is the conclusion and future research direction.

2. Genetic algorithm

The genetic algorithm (Mirjalili et al., 2019) models reproduction and genes in nature. The algorithm works with a population of solutions. It performs small or large changes on the populations. It uses operations such as crossover and mutation. During crossover, two new individuals are created from two parent individuals. A mutation is a small change in a single individual. In the paper, the 2-opt (Englert et al., 2014) operator acts as mutation. The paper investigates the following crossover methods: Order Crossover (Arram et al., 2019), Cycle Crossover (Hussain et al., 2017), and Partially Matched Crossover (Ahmed et al., 2016).

The Genetic Algorithm consists of the following steps:

1. Initialization of parameters and the population
2. Evaluation of population elements
3. Creating a new population
   a. Elitism: certain elements are transferred unchanged into the population
   b. Crossing of selected parents
   c. Mutation of the children
4. Continue step 2–3. until the stop condition is not met

2.1. 2-opt operator

The 2-opt (Englert et al., 2014) operator means exchanging the elements of a section of a permutation. It is often used in permutation problems because it is easy-to-implement and efficient operator.

The 2-opt algorithm performs the 2-opt step iteratively. The algorithm is often used to solve permutation problems, especially the Traveling Salesman Problem (TSP). The algorithm performs the 2-opt step until it improves the solution.

2.2. Order Crossover

The order crossover (OX) (Arram et al., 2019) operator is commonly used in permutation problems. The algorithm ensures that the child elements are also permutations.

The first step of the algorithm involves selecting two parents. Then, a random section in the two parents is chosen, determining the segment that will be passed from one parent to the child. The algorithm copies the selected segment of one parent to the same positions in the offspring chromosome. Subsequently, it fills in the empty spaces of the offspring's chromosome with the genes of the other parent's chromosome, maintaining the original order, starting after the first crossover point. During the OX steps, the algorithm copies a segment from one parent to the offspring, and then fills the remaining spaces with genes from the other parent, preserving the order of the elements.

2.3. Cycle Crossover

Cycle crossover (CX) (Hussain et al., 2017) is a special crossover operator used in permutation problems. This operator ensures that the children's solutions will also be permutations.

The algorithm starts by selecting two parents. Then, it identifies cycles on the parents. A cycle is formed when the algorithm begins at a position and then follows the elements from one parent to the next until it returns to the original element. The algorithm then copies the cycles to the offspring in alternating order: one offspring takes the elements of the cycle from one parent, while the other offspring takes the elements from the other parent.

2.4. Partially Matched Crossover

The Partially Matched Crossover (PMX) (Ahmed et al., 2016) is a special crossover operator mainly used in permutation problems. It involves two parent chromosomes and two randomly selected crossover points. These points define a segment within which genes are exchanged. The segment of the first parent is placed in the corresponding location of the child of the second parent, and vice versa. Genes outside the segment are rearranged to ensure no duplications and each gene appears exactly once. The empty spaces in the child chromosome are filled with genes from the other parent, based on the fitting section.

While PMX efficiently handles permutation problems, it is more complex and time-consuming than some other crossover operators. Additionally, finding optimal crossover points can be challenging.

3. Test results

This section contains the test results. The article performed the test runs on the Taillard dataset (Taillard, 1993), from Ta001 to Ta030. I also compared the effectiveness of the Genetic Algorithm with test results of other metaheuristic algorithms, namely HMM-PFA, HGA, IIGA, DSOMA, HGSA, IWO. The table also shows the relative performances. In the case of IWO, there were only test results for the following data sets: Ta001, Ta011, Ta021.

Table 1. *Test results*

		Relative performance					
Instance	GA	HMM-PFA (Qu et al., 2018) %	HGA (Wei et al., 2018) %	IIGA (Wei et al., 2018) %	DSOMA (Wei et al., 2018) %	HGSA (Wei et al., 2018) %	IWO (Zhou et al., 2014) %
Ta001	1297	114.57	111.72	114.57	105.94	102.08	107.09
Ta002	1360	112.35	107.35	112.35	103.53	106.03	–
Ta003	1167	125.11	118.77	125.11	109.68	94.1	–
Ta004	1325	119.85	114.79	119.85	109.28	110.87	–
Ta005	1250	115.92	112.24	115.92	107.28	103.28	–
Ta006	1218	121.59	117.41	121.59	111.90	114.20	–
Ta007	1251	118.55	116.79	118.55	110.39	103.84	–
Ta008	1217	121.77	117.75	121.77	113.31	106.16	–
Ta009	1274	115.31	109.73	115.31	107.77	102.51	–
Ta010	1174	117.29	112.78	117.29	109.28	105.03	–
Ta011	1643	124.41	118.99	122.40	103.35	104.26	134.33
Ta012	1754	123.49	121.04	123.49	104.50	97.95	–
Ta013	1564	124.04	122.25	124.04	107.16	99.43	–
Ta014	1446	125.24	123.24	125.24	106.92	104.84	–
Ta015	1491	129.64	129.64	129.64	108.45	105.50	–
Ta016	1497	126.39	122.04	126.39	106.21	97.33	–
Ta017	1585	123.85	122.65	123.85	102.33	102.33	–
Ta018	1630	126.20	123.07	126.20	106.20	107.30	–
Ta019	1636	120.60	116.63	120.60	106.78	99.27	–
Ta020	1738	118.01	115.13	118.01	102.53	99.08	–
Ta021	2391	124.34	121.79	124.34	101.88	97.49	134.92
Ta022	2180	130.83	127.52	118.44	102.48	104.59	–
Ta023	2464	122.28	118.59	122.28	100.61	100.65	–

Ta024	2325	129.08	127.61	129.08	100.99	101.59	–
Ta025	2446	122.77	120.73	122.77	99.45	102.49	–
Ta026	2323	129.06	125.18	128.63	102.58	102.24	–
Ta027	2333	130.82	127.30	130.82	102.44	100.34	–
Ta028	2328	121.95	118.69	121.95	100.00	97.9	–
Ta029	2354	127.82	126.25	127.82	100.38	102.38	–
Ta030	2209	134.86	132.14	134.86	105.16	108.69	–

The test results are the followings: for Ta001, GA gave a fitness value of 1297. It was 14% better than HMM-PFA, 11% better than HGA, 14% better than IIGA, 5% better than DSOMA, 2% better than HGSA and 7% better like the IWO. GA gave a fitness value of 1360 for Ta002. It was 12% better than HMM-PFA, 7% better than HGA, 12% better than IIGA, 3% better than DSOMA and 6% better than HGSA. The Genetic Algorithm gave Ta003 a fitness value of 1167. This value was 25% better than HMM-PFA, 18% better than HGA, 25% better than IIGA, 9% better than DSOMA and 5% worse. like HGSA. The Genetic Algorithm gave 1325 fitness results for Ta004. It was 19% better than HMM-PFA, 14% better than HGA, 19% better than IIGA, 9% better than DSOMA, 10% better than HGSA. The algorithm gave 1250 fitness results for Ta005. This result was 15% better than HMM-PFA, 12% better than HGA, 15% better than IIGA, 7% better than DSOMA, and 3% better than the HGSA.

For a larger dataset, for example, Ta021, the Genetic algorithm gave 2391 fitness result which was 24% better than HMM-PFA, 21% better than HGA, 24% better than IIGA, 1% better than DSOMA, 2% worse than HGSAm and 34% better than IWO.

Table 2. *Test results comparisons*

Algorithm	Number of data rows (on which the comparisons were created)	Number of better results
HMM-PFA	30	30
HGA	30	30
IIGA	30	30
DSOMA	30	28
HGSA	30	22
IWO	3	3

The test result comparisons of the Genetic algorithm are as follows: it gave better results than HMM-PFA, HGA, IIGA, and IWO algorithms in all cases, and gave better results in most cases compared to the DSOMA and HGSA algorithms.

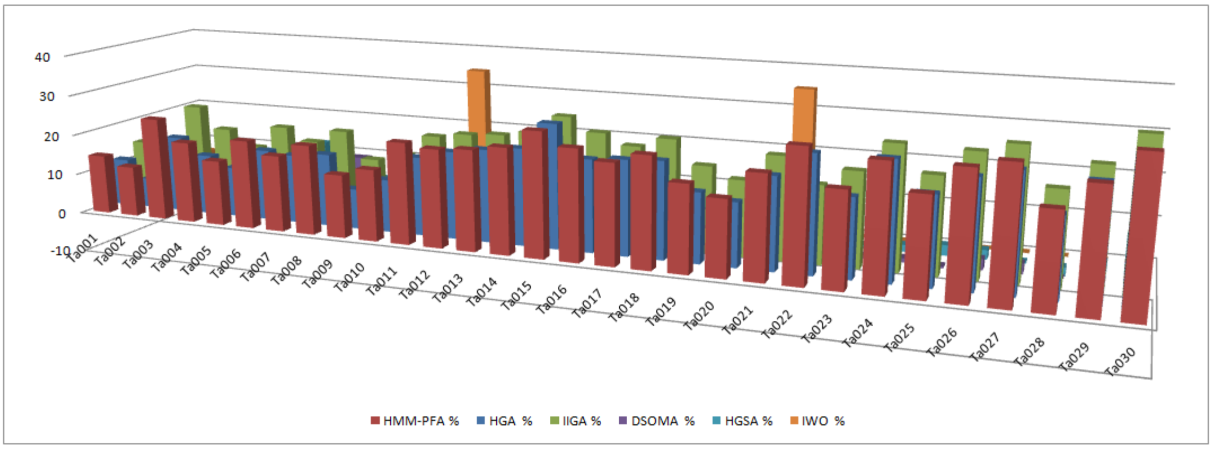

Figure 2. *Genetic Algorithm result (vertical bar chart)*

Figure 2 presents a multidimensional vertical bar chart containing the results of each comparison algorithm. The x-axis represents the names of the benchmark dataset, the y-axis represents the percentage differences taken from the genetic algorithm, and the colors represent the comparison algorithms.

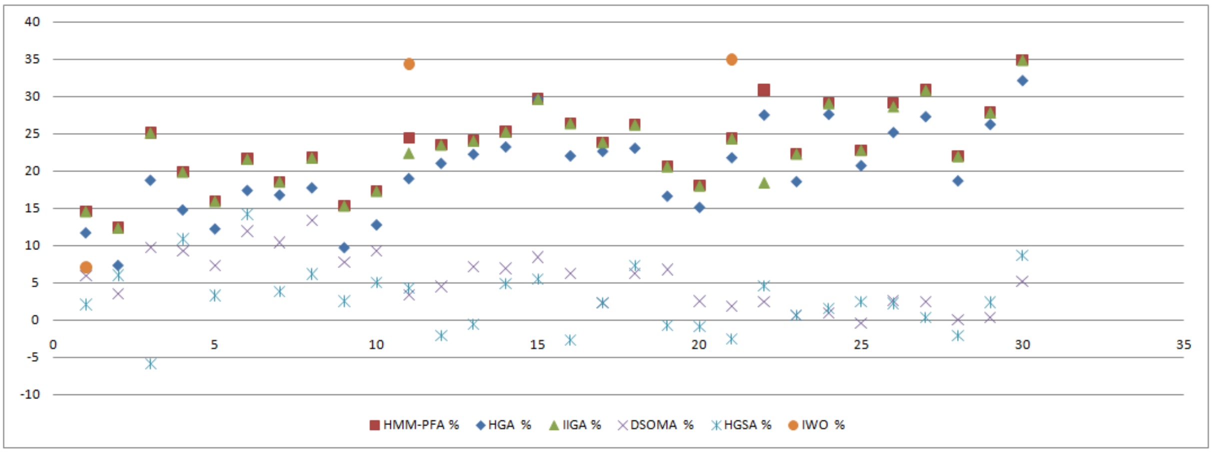

Figure 3. *Genetic Algorithm result (scatter chart)*

The figure below contains the results of the table in a scatter chart. The colors and shapes of the individual points represent the different algorithms, the x-axis represents the benchmark dataset, and the y-axis is the percentage by which the given algorithm is worse or better than the Genetic Algorithm.

It can be seen from the diagram that for most algorithms the Genetic Algorithm was at least 5-10% better, but in many cases, the GA was 20-25% better.

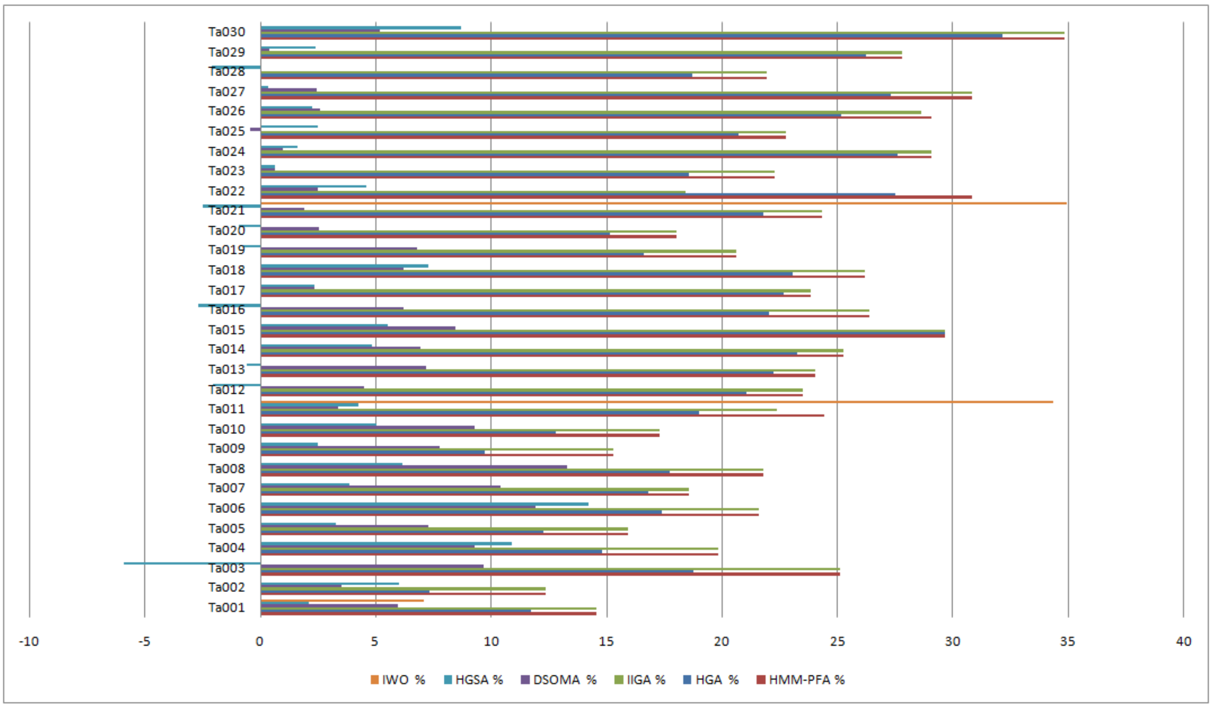

Figure 4. *Genetic Algorithm result (horizontal bar chart)*

The figure below shows the results in a bar chart, where the individual colors represent the individual algorithms, the x-axis the percentage values, and the y-axis the benchmark datasets.

4. Conclusion and future research

This article presents the solution of a production scheduling task, the Flow Shop Scheduling problem, with a Genetic Algorithm. After the literature review, the FSS and GA were adetailed in the article. Then the test results were presented. Here, the efficiency of the algorithm against six other algorithms is presented in tabular form. In addition, the test results were represented with three types of diagrams. Based on the test results, GA gave better results than the other algorithms in the majority of cases. A future research area could be the solution of FSS with other heuristics, for example, the Firefly Algorithm, Simulated Annealing, Memetic Algorithm, etc.

5. Acknowledgements

„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."

Új Nemzeti Kiválósági Program

MINISTRY OF CULTURE AND INNOVATION

References

[1] Kulcsár, G., & Erdélyi, F. (2007). A new approach to solve multi-objective scheduling and rescheduling tasks. *International Journal of Computational Intelligence Research*, 3 (4), 343–351. **https://doi.org/10.5019/j.ijcir.2007.115**

[2] Framinan, J. M., Gupta, J. N., & Leisten, R. (2004). A review and classification of heuristics for permutation flow-shop scheduling with makespan objective. *Journal of the Operational Research Society*, 55, 1243–1255. **https://doi.org/10.1057/palgrave.jors.2601784**

[3] de Jong, A. W., Rubrico, J. I., Adachi, M., Nakamura, T., & Ota, J. (2017, August). Big data in automation: Towards generalized makespan estimation in shop scheduling problems. In *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*, IEEE, 1516–1521. **https://doi.org/10.1109/COASE.2017.8256319**

[4] Anand, E., & Panneerselvam, R. (2015). Literature review of open shop scheduling problems. *Intelligent Information Management*, 7 (01), 33. **https://doi.org/10.4236/iim.2015.71004**

[5] Mellor, P. (1966). A review of job shop scheduling. *Journal of the Operational Research Society*, 17 (2), 161–171. **https://doi.org/10.1057/jors.1966.24**

[6] Yadav, A., & Jayswal, S. C. (2018). Modelling of flexible manufacturing system: a review. *International Journal of Production Research*, 56 (7), 2464–2487. **https://doi.org/10.1080/00207543.2017.1387302**

[7] Ruiz, R., & Vázquez-Rodríguez, J. A. (2010). The hybrid flow shop scheduling problem. *European Journal of Operational Research*, 205 (1), 1–18. **https://doi.org/10.1016/j.ejor.2009.09.024**

[8] Jiang, S. L., & Zhang, L. (2019). Energy-oriented scheduling for hybrid flow shop with limited buffers through efficient multi-objective optimization. *IEEE Access*, 7, 34477–34487. **https://doi.org/10.1109/ACCESS.2019.2904848**

[9] Nowicki, E., & Smutnicki, C. (1998). The flow shop with parallel machines: A tabu search approach. *European Journal of Operational Research*, 106 (2–3), 226–253. **https://doi.org/10.1016/S0377-2217(97)00260-9**

[10] Javadian, N., Amiri-Aref, M., Hadighi, A., Kazemi, M., & Moradi, A. (2010). Flexible flow shop with sequence-dependent setup times and machine availability constraints. *International Journal of Management Science and Engineering Management*, 5 (3), 219–226. **https://doi.org/10.1080/17509653.2010.10671111**

[11] Oukil, A., & El-Bouri, A. (2021). Ranking dispatching rules in multi-objective dynamic flow shop scheduling: a multi-faceted perspective. *International Journal of Production Research*, 59 (2), 388–411. **https://doi.org/10.1080/00207543.2019.1696487**

[12] Wang, L., Pan, Q. K., & Tasgetiren, M. F. (2010). Minimizing the total flow time in a flow shop with blocking by using hybrid harmony search algorithms. *Expert Systems with Applications*, 37 (12), 7929–7936. **https://doi.org/10.1016/j.eswa.2010.04.042**

[13] Chen, K., Li, D., & Wang, X. (2020). Makespan minimization in two-machine flow-shop scheduling under no-wait and deterministic unavailable interval constraints. *Journal of Systems Science and Systems Engineering*, 29, 400–411. **https://doi.org/10.1007/s11518-020-5456-2**

[14] Mirsanei, H. S., Karimi, B., & Jolai, F. (2009). Flow shop scheduling with two batch processing machines and nonidentical job sizes. *The International Journal of Advanced Manufacturing Technology*, 45, 553–572. **https://doi.org/10.1007/s00170-009-1986-y**

[15] Huo, Y., & Huang, J. X. (2016, May). Parallel ant colony optimization for flow shop scheduling subject to limited machine availability. In *2016 IEEE International Parallel and Distributed Processing Symposium Workshops* (IPDPSW), IEEE, 756–765. **https://doi.org/10.1109/IPDPSW.2016.151**

[16] Chamnanlor, C. (2013). Heuristics for scheduling hybrid flow shop with time windows. *International Journal of Engineering and Technology*, 5 (1), 41–44. **https://doi.org/10.7763/IJET.2013.V5.506**

[17] Jiang, S. L., & Zhang, L. (2019). Energy-oriented scheduling for hybrid flow shop with limited buffers through efficient multi-objective optimization. *IEEE Access*, 7, 34477–34487. **https://doi.org/10.1109/ACCESS.2019.2904848**

[18] Neufeld, J. S., Schulz, S., & Buscher, U. (2023). A systematic review of multi-objective hybrid flow shop scheduling. *European Journal of Operational Research*, 309 (1), 1–23. **https://doi.org/10.1016/j.ejor.2022.08.009**

[19] Kouvelis, P., Daniels, R. L., & Vairaktarakis, G. (2000). Robust scheduling of a two-machine flow shop with uncertain processing times. *Iie Transactions*, 32 (5), 421–432. **https://doi.org/10.1023/A:1007640726040**

[20] Azzouz, A., Ennigrou, M., & Ben Said, L. (2018). Scheduling problems under learning effects: classification and cartography. *International Journal of Production Research*, 56 (4), 1642–1661. **https://doi.org/10.1080/00207543.2017.1355576**

[21] Kis, T., Drótos, M., Erdős, G., & Kovács, A. (2006). The interval flow shop model. In: *Proceedings of PMS*, 26–28.

[22] Emmons, H., Vairaktarakis, G., Emmons, H., & Vairaktarakis, G. (2013). Flexible Flow Shops. Flow Shop Scheduling: Theoretical Results, Algorithms, and Applications, 247–268. **https://doi.org/10.1007/978-1-4614-5152-5_8**

[23] Li, D., Meng, X., Liang, Q., & Zhao, J. (2015). A heuristic-search genetic algorithm for multi-stage hybrid flow shop scheduling with single processing machines and batch processing machines. *Journal of Intelligent Manufacturing*, 26, 873–890. **https://doi.org/10.1007/s10845-014-0874-y**

[24] Xiao, Y. Y., Zhang, R. Q., Zhao, Q. H., & Kaku, I. (2012). Permutation flow shop scheduling with order acceptance and weighted tardiness. *Applied Mathematics and Computation*, 218 (15), 7911–7926. **https://doi.org/10.1016/j.amc.2012.01.073**

[25] Burduk, A., Musiał, K., Kochańska, J., Górnicka, D., & Stetsenko, A. (2019). Tabu search and genetic algorithm for production process scheduling problem. *LogForum*, 15 (2), 181–189. **http://doi.org/10.17270/J.LOG.2019.315**

[26] Gubán, M., & Gubán, Á. (2012). Production scheduling with genetic algorithm. *Advanced Logistic Systems: Theory and Practice*, 6 (1), 33–44.

[27] Jawahar, N., Aravindan, P., & Ponnambalam, S. G. (1998). A genetic algorithm for scheduling flexible manufacturing systems. *The International Journal of Advanced Manufacturing Technology*, 14, 588–607. **https://doi.org/10.1007/BF01301703**

[28] Fang, H. L. (1994). *Genetic algorithms in timetabling and scheduling*. Doctoral dissertation, University of Edinburgh.

[29] Hegazy, T., & Kassab, M. (2003). Resource optimization using combined simulation and genetic algorithms. *Journal of Construction Engineering and Management*, 129 (6), 698–705. **https://doi.org/10.1061/(ASCE)0733-9364(2003)129:6(698)**

[30] Dasgupta, K., Mandal, B., Dutta, P., Mandal, J. K., & Dam, S. (2013). A genetic algorithm (ga) based load balancing strategy for cloud computing. *Procedia Technology*, 10, 340–347. **https://doi.org/10.1016/j.protcy.2013.12.369**

[31] Pellazar, M. B. (1994, May). Vehicle route planning with constraints using genetic algorithms. In *Proceedings of National Aerospace and Electronics Conference (NAECON'94)*, IEEE, 111–118. **https://doi.org/10.1109/NAECON.1994.333010**

[32] Chakroborty, P., Deb, K., & Sharma, R. K. (2001). Optimal fleet size distribution and scheduling of transit systems using genetic algorithms. *Transportation Planning and Technology*, 24 (3), 209–225. **https://doi.org/10.1080/03081060108717668**

[33] Chouhan, K. S., & Deulkar, V. (2019). A review article of transportation vehicles routing scheduling using genetic algorithm. *Int J Sci Res Eng Trends*, 2 (5).

[34] Kubota, N., Fukuda, T., Shibata, S., Yamamoto, H., & Inada, Y. (1997, November). Intelligent manufacturing system by the evolutionary computation-scheduling and planning of the flexible transportation. In *Proceedings of the IECON'97 23rd International Conference on Industrial Electronics, Control, and Instrumentation (Cat. No. 97CH36066)*, IEEE, Vol. 3, 1136–1141. **https://doi.org/10.1109/IECON.1997.668446**

[35] Oudani, M., El Hilali Alaoui, A., & Boukachour, J. (2014). An efficient genetic algorithm to solve the intermodal terminal location problem. *International Journal of Supply and Operations Management*, 1 (3), 279–296. **https://dx.doi.org/10.22034/2014.3.02**

[36] Xu, G., Monsalve-Serrano, J., Jia, M., & García, A. (2020). Computational optimization of the dual-mode dual-fuel concept through genetic algorithm at different engine loads. *Energy Conversion and Management*, 208, 112577. **https://doi.org/10.1016/j.enconman.2020.112577**

[37] Zhang, L., Gao, Y., Sun, Y., Fei, T., & Wang, Y. (2019). Application on cold chain logistics routing optimization based on improved genetic algorithm. *Automatic Control and Computer Sciences*, 53, 169–180. **https://doi.org/10.3103/S0146411619020032**

[38] Aszemi, N. M., & Dominic, P. D. D. (2019). Hyperparameter optimization in convolutional neural network using genetic algorithms. *International Journal of Advanced Computer Science and Applications*, 10 (6). **https://doi.org/10.14569/IJACSA.2019.0100638**

[39] Han, S. S., & May, G. S. (1996, November). Optimization of neural network structure and learning parameters using genetic algorithms. In *Proceedings Eighth IEEE International Conference on Tools with Artificial Intelligence*, IEEE, 200–206. **https://doi.org/10.1109/TAI.1996.560452**

[40] Rojas, M. G., Olivera, A. C., & Vidal, P. J. (2022). Optimising Multilayer Perceptron weights and biases through a Cellular Genetic Algorithm for medical data classification. *Array*, 14, 100173. **https://doi.org/10.1016/j.array.2022.100173**

[41] Bevilacqua, V., Mastronardi, G., Menolascina, F., Pannarale, P., & Pedone, A. (2006, July). A novel multi-objective genetic algorithm approach to artificial neural network topology optimisation: the breast cancer classification problem. In *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, IEEE, 1958–1965. **https://doi.org/10.1109/IJCNN.2006.246940**

[42] Takahashi, H., Agui, T., & Nagahashi, H. (1993, August). Designing adaptive neural network architectures and their learning parameters using genetic algorithms. In *Science of Artificial Neural Networks II*, Vol. 1966, SPIE, 208–215. **https://doi.org/10.1117/12.152652**

[43] Pellazar, M. B. (1994, May). Vehicle route planning with constraints using genetic algorithms. In *Proceedings of National Aerospace and Electronics Conference* (NAECON'94), IEEE, 111–118. **https://doi.org/10.1109/NAECON.1994.333010**

[44] Kordos, M., Boryczko, J., Blachnik, M., & Golak, S. (2020). Optimization of warehouse operations with genetic algorithms. *Applied Sciences*, 10 (14), 4817. **https://doi.org/10.3390/app10144817**

[45] Radhakrishnan, P., Prasad, V. M., & Gopalan, M. R. (2009). Inventory optimization in supply chain management using genetic algorithm. *International Journal of Computer Science and Network Security*, 9 (1), 33–40.

[46] Ongcunaruk, W., Ongkunaruk, P., & Janssens, G. K. (2021). Genetic algorithm for a delivery problem with mixed time windows. *Computers & Industrial Engineering*, 159, 107478. **https://doi.org/10.1016/j.cie.2021.107478**

[47] East, A. R., & Smale, N. J. (2008). Combining a hybrid genetic algorithm and a heat transfer model to optimise an insulated box for use in the transport of perishables. *Vaccine*, 26 (10), 1322–1334. **https://doi.org/10.1016/j.vaccine.2007.12.055**

[48] Gupta, R. K., & Bhunia, A. K. (2006). An application of real-coded Genetic Algorithm (RCGA) for integer linear programming in Production-Transportation Problems with flexible transportation cost. *AMO-Adv Model Optim*, 8 (1).

[49] Ko, H. J., & Evans, G. W. (2007). A genetic algorithm-based heuristic for the dynamic integrated forward/reverse logistics network for 3PLs. *Computers & Operations Research*, 34 (2), 346–366. **https://doi.org/10.1016/j.cor.2005.03.004**

[50] Bo, Z. W., Hua, L. Z., & Yu, Z. G. (2006). Optimization of process route by genetic algorithms. *Robotics and Computer-Integrated Manufacturing*, 22 (2), 180–188. **https://doi.org/10.1016/j.rcim.2005.04.001**

[51] Mirjalili, S., & Mirjalili, S. (2019). Genetic algorithm. *Evolutionary Algorithms and Neural Networks: Theory and Applications*, 43–55. **https://doi.org/10.1007/978-3-319-93025-1**

[52] Englert, M., Röglin, H., & Vöcking, B. (2014). Worst case and probabilistic analysis of the 2-Opt algorithm for the TSP. *Algorithmica*, 68 (1), 190–264. **https://doi.org/10.1007/s00453-013-9801-4**

[53] Arram, A., & Ayob, M. (2019). A novel multi-parent order crossover in genetic algorithm for combinatorial optimization problems. *Computers & Industrial Engineering*, 133, 267–274. **https://doi.org/10.1016/j.cie.2019.05.012**

[54] Hussain, A., Muhammad, Y. S., Nauman Sajid, M., Hussain, I., Mohamd Shoukry, A., & Gani, S. (2017). Genetic algorithm for traveling salesman problem with modified cycle crossover operator. *Computational Intelligence and Neuroscience, 2017*. **https://doi.org/10.1155/2017/7430125**

[55] Ahmed, Y. E., Adjallah, K. H., Stock, R., & Babikier, S. F. (2016). Wireless sensor network lifespan optimization with simple, rotated, order and modified partially matched crossover genetic algorithms. *IFAC-PapersOnLine*, 49 (25), 182–187. **https://doi.org/10.1016/j.ifacol.2016.12.031**

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

[57] 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*. **https://doi.org/10.1155/2018/1973604**

[58] 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**

[59] 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**