Review Article
Araştırma Makalesi

Bartın University International Journal of Natural and Applied Sciences
JONAS, 5(2): 86-94 e-ISSN: 2667-5048
Aralık/December, 2022 DOI: 10.55930/jonas.1121763

OPTIMIZING THE PERMUTATION FLOWSHOP SCHEDULING PROBLEM (PFSP) USING THE SCATTER SEARCH METHOD

Uğur Sinan Eren[1], Ezgi Güler[2], Yıldız Şahin[3*],
[1] Kocaeli University, Inst. of Natural and Applied Science, Industrial Engineering, Kocaeli, Turkey.
[2] Bilecik Şeyh Edebali University, Inst. of Natural and Applied Science, Industrial Eng., Bilecik, Turkey.
[3] Kocaeli University, Faculty of Engineering, Industrial Engineering, Kocaeli, Turkey.

Abstract

Scheduling is the process of optimizing limited resources, depending on the objectives. Scheduling problems are one of the decision-making problems that play a critical role in production and service systems. Continuing production regularly and systematically is an important issue for production planners. Permutation flow shop scheduling, which is a sub-branch of production scheduling, is defined as "n" jobs being processed simultaneously on "m" machines. Permutation Flow Shop Scheduling Problems (PFSPs) are in the complex and difficult problem class. Many metaheuristic methods have been proposed to solve such problems. In this study, the Scatter Search method, which is one of the population-based evolutionary methods of metaheuristic methods, was used to solve the Permutation Flow Shop Scheduling Problem (PFSP). The scatter search method was analyzed with the algorithm prepared on JavaScript programming language. With the scatter search, the total completion time of the jobs was minimized and the effectiveness of the method was tested on the problem groups frequently used in the literature. The use of the JavaScript programming language in this study has contributed to the literature on testing large-scale problems. The distribution search algorithm has a positive effect on the PTSP with an average of 2% difference from the best-known solutions due to the minimization of work times.

Key Words: Scheduling problem, flow shop scheduling, metaheuristic method, scatter search, JavaScript.

1. Introduction

Today, the rapid progress of technology and science causes the formation of a competitive market environment for businesses. In this competitive environment, delivering the product or service within the time given to the customer is an important issue in terms of customer satisfaction. The first step in establishing a dynamic business system that can operate at the desired capacity is an appropriate scheduling (Mete, 2019). Scheduling deals with the allocation of resources or jobs to machines over time in order to optimize a specific goal (Arshad et al., 2021). Scheduling problems in manufacturing are in the category of NP-complete problems. The general purpose in production scheduling is to ensure that the jobs planned to be done with the machines at hand are completed in the most appropriate job order in the minimum time. Scheduling problems involving a large number of jobs and machines are included in the combined optimization problems because the solution space grows exponentially. The difficulty of the scheduling problem increases as the constraints such as the delay criterion and the concurrent job criterion in the production structure increase (Kaya et al., 2020).

Metaheuristic methods used for NP-complete problems where mathematical modeling cannot be done or the cost of establishing a mathematical model is high are frequently preferred by decision makers because they have good computational power and allow model development with satisfactory results (Kaya & Fığlalı, 2018). The main purpose of metaheuristic methods is to search the search space and to produce near-optimal solutions without getting stuck with the local optimum (Osman & Laporte, 1996). Metaheuristic methods are examined in three groups as physics-based, local search-based and population-based metaheuristics. Big Crunch Optimization, Atom Search Optimization, Ray Optimization are some methods of physics-based metaheuristics (Abdollahzadeh et al., 2021). Simulated Annealing, Iterated Local Search, and Tabu Search algorithms can be given as examples of local search-based methods (Erol, 2006). Population-based algorithms include evolutionary algorithms such as Particle Swarm Optimization, Ant Colony Optimization, Genetic Algorithm, and Scatter Search (Osman & Kelly, 1996).

***Sorumlu Yazar (Corresponding Author):**
Yıldız ŞAHİN; Kocaeli University, Faculty of Engineering,
Industrial Engineering,
Kocaeli, Turkey.

Geliş (Received) : 26.05.2022
Kabul (Accepted) : 19.09.2022
Basım (Published) : 31.12.2022

The scatter search method, which is one of the evolutionary algorithms, consists of strategies that generate composite decision rules and constraints (Oktay & Engin, 2006). The scatter method basically aims to create new solutions by differentiating reference set solutions with combinations and is widely used in the solution of flow shop problems, which are combinatorial optimization problems. There are many studies in the literature in which the scatter search method is used alone or in combination with other metaheuristic methods for the flow shop scheduling problem. Some of these studies are summarized.

Nowicki & Smutnicki (2006) developed a mixed method in the flow scheduling problem based on the scatter search and the neighbor concept in tabu search. They tried their proposed method on 30 large-scale difficult test problems and obtained better results than known for 20 of them. Rahimi-Vahed et al. (2008) studied a two-criteria no-wait flow scheduling problem in which the weighted average completion time and delay times of jobs are minimized simultaneously. They proposed a new multi-objective scatter search algorithm. Test problems are solved to prove the effectiveness of the proposed algorithm. The reliability of the proposed algorithm is compared with a multi-objective genetic algorithm (GA). Computational results showed that the proposed algorithm outperforms GA, especially for large-sized problems (Rahimi-Vahed et al., 2008). Saravanan & Haq investigated the scheduling problem in flexible manufacturing systems. They determined the objective function as minimizing the total work time and the total penalty costs at the same time. A scatter search metaheuristic is proposed for problem solving. The results obtained from the test problems are compared with metaheuristic methods such as genetic algorithm, particle swarm optimization and annealing simulation (Saravanan & Haq, 2008). Moghaddam et al. (2010), proposed a new mathematical model for the scheduling problem in a business system where parts can visit different cells. Simultaneously, they aimed to minimize setup costs related to brand, intracellular movement, delay, and sequence. Due to the complexity of the problem, they used a scatter search-based metaheuristic. They compared the results of 10 test problems according to the completion times of the jobs. Sadiq & Muhamad (2012) proposed a scatter search algorithm for the flow shop scheduling problem in their study. In the study, they determined the objective function as minimizing the maximum completion time of the jobs. They searched for random solutions with the proposed algorithm and improved the distribution search of all solutions by applying the idea of all machines working at the same time. Yang et al. (2017) developed a distributed search-based model for the distributed assembly flow shop scheduling problem that considers machine and job characteristics. The model is based on 10 small and 5 large-scale test problems. They were compared to 6 heuristics and found that scatter search outperformed all of them. Pan et al. (2019), dealt with the distributed PFSP in their work. They determined the objective function of the problem as the total flow time. They compared the metaheuristic methods of discrete artificial bee colony, scatter search, and iterated local search on test problems. Abdelmaguid (2020) developed a two-neighbor search and a solution aggregation function in a multi-process open shop-type scheduling problem consisting of non-identical machines and used route-joining scatter search algorithm. In his study, he made parameter analysis and tried the algorithm in test problems. It has been seen that the developed algorithm has the capacity to obtain optimal and near-optimal results. Külahlı et al. (2021) proposed a new hybrid scatter search algorithm for flexible workshop scheduling problems. They determined the objective function in the proposed method as minimizing the maximum completion times for flexible workshop scheduling problems. A full factorial experimental design was made for the performance parameters. Behnamian et al. (2021) considered a two-objective flexible scheduling problem with independent setup time. The objective functions in the problems are determined as minimizing the maximum completion time of the works and the total delay. After proposing a mixed integer nonlinear programming model, a scatter search algorithm is developed to obtain near-optimal solutions. Obtained results were compared with Non-Dominated Sorting Genetic Algorithm II (NSGA-II) over few samples flexible scheduling problems.

2. Material and Methods

In this study, the optimization of the PFSP with scatter search is investigated. Different problems and methods in the literature are used for the steps of the scatter search algorithm used in the application. Angular framework is used with JavaScript in optimization model design. In the next subsections, PFSP and scatter search methodology are given.

2.1. PFSP

Production scheduling can be defined as the activity of assigning appropriate production resources to jobs in order to meet the desired criteria (Graves, 1981). In order to optimize the desired criteria, the specified jobs must be put in order. The job sequencing method ensures that the jobs are ranked in a way to be completed in the shortest time in total or the total of lost times is minimized by passing through certain processes or machines in a production

department (Osman & Laporte, 1996). Flow shop scheduling is known as one of the most basic of classical scheduling problems. Graves (1981), who classified production scheduling in five dimensions, examined the process complexity dimension in four different parts according to the number of steps. Graves (1981) defined flow-type scheduling problems as multi-stage problems in the dimension of transaction complexity.

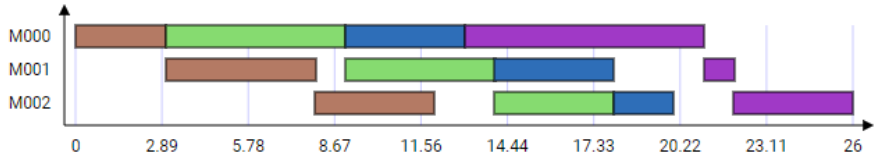

Fig. 1. An example flow shop schedule

The flow shop scheduling problem finds the ordering of jobs for each machine according to a certain performance criterion. Technological constraints may require jobs to be processed through the same machine queue. It is also assumed that in most cases the rank of jobs for each machine is the same, with n jobs being processed in the same order on m machines (Fink & Voß, 2003) as in Figure 1. In flow shop scheduling problems, there are two different types of scheduling: schedules in which the job sequences are different on each machine and permutation schedules where the job sequences are the same on each machine. In the schedules where the job sequences are different n! for each machine different work sequences are obtained, and the number of possible schedules for m machines is (n!)m. In permutation schedules where the job sequences are the same on every machine, the possible number of schedules decreases to (n!) (Baskar & Xavior, 2021).

2.2. Scatter Search Algorithm

Scatter search algorithm was first proposed by Glover in 1977 as a metaheuristic method for solving integer programming problems and designed as a method to relax constraints (Marti et al., 2006). Scatter search differs from other evolutionary methods in that it produces new results by systematically selecting multiple solutions from the reference set. A new solution set is created by selecting two or more subsets from the solutions in the said set (Çiçekli & Bozkurt, 2016; Sagarna & Lozano, 2006). Scatter search algorithm consists of 5 essential methods and various implementations to these methods exist. The algorithm methods (Marti et al., 2006) and the approach we adopted is as follows:

1. A Diversification Method to generate a collection of diverse trial solutions, using an arbitrary trial solution (or seed solution) as an input.
2. An Improvement Method to transform a trial solution into one or more enhanced trial solutions.
3. Reference Set Update Method to build and maintain a reference set consisting of the good and diverse solutions found bounded by the reference set size parameters.
4. Subset Generation Method to operate on the reference set, to produce a subset of its solutions as a basis for creating combined solutions.
5. Solution Combination Method to transform a given subset of solutions produced by the Subset Generation Method into a combined solution.

Scatter search algorithm initializes the solution population via diversification method. In this implementation; CDS (Campbell et al., 1970; Mashuri et al., 2019), NEH (Çiçekli & Bozkurt, 2016; Alharkan, 2005; Nawaz et al., 1983), Palmer (Alharkan, 2005; Palmer, 1965) and SPT (Çiçekli & Bozkurt, 2016) algorithms are used to create good enough solutions as a starting point. Rest of the solutions in the population are constructed randomly. Then each solution in population is subjected to the improvement method, which is chosen as a local search algorithm that swaps each job with its neighboring jobs in the solution then updates the solution when an improvement is made (Stützle & Hoos, 2000). Reference set update method adds select number of best and worst solutions based on their makespan to the reference set. Then each combination of good and diverse reference sets is made subsets. Next each solution duo is combined using the path relinking algorithm (Riahi et al., 2017) and each offspring solution is then subjected to improvement method subsequently. As given in Figure 2, algorithm returns back to updating reference set until the ending criteria met.

It is intended to create a new variant of scatter search algorithm by the means of fusing together several popular sub methods of scatter search that are used in the literature. A parameter analysis aimed at minimizing makespan is also made.

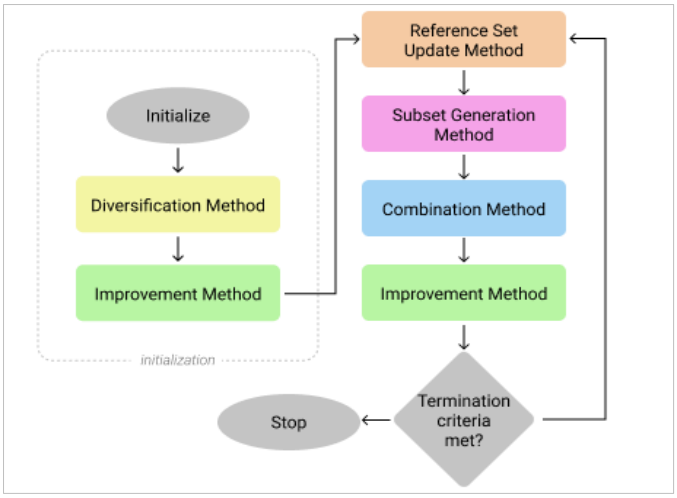

Fig. 2. Scatter search algorithm flow

2.3. Data Collection

Performance data for parameter analysis is gathered by testing the problem Tai019 consisting of 10 machines and 20 jobs. By keeping the problem size small it is ensured to gather more data in shorter time. Reason for picking Tai019 vs. other similar sized problems is that makespan of Tai019 had a standard deviation of 12 while the average of other problems was 2.

Iteration number, population size, reference set size, good & diverse reference set sizes and their effect on makespan and algorithm runtime were analyzed using the 3000 data points that were gathered. Parameters are generated as follows and as shown in Table 1:

Tab. 1. Parameter generation ranges

Parameter	**Generation Range**		
Iteration	20	–	60
Population size	20	–	200
Reference set size	1%	–	100%
Good reference set size	0%	–	100%
Diverse reference set size	100%	–	0%

- Iteration: 1 – 60. 500 separate data points in range of 0 – 500 were generated to examine iteration number while keeping other parameters set according to Riahi's study (2017). Effect on makespan is not statistically significant despite having a correlation as seen in Figure 3, p and r values are 0.098 and - 0.07 respectively.
- Population size: 20 – 200. Rimli et al. (2017) recommends that diverse reference size should be 10 times bigger than the problem size. To ensure this, population size is generated between being equal to the problem size and it's tenfold, since diverse references are a subset of the population.
- Reference set size: 0% - 100% of the population size.
- Good and diverse reference set sizes: 0% - 100% of the reference set size, given their sum is 1.

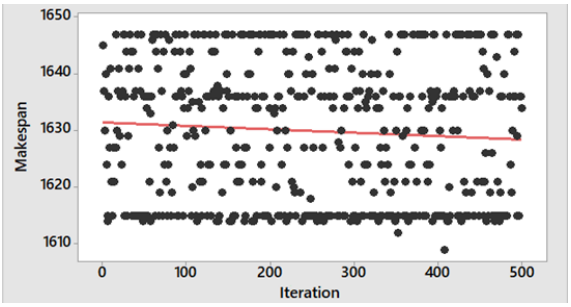

Fig. 3: Relation between makespan and iteration count

3. Results

A total of 3000 test runs are done using the randomly generated parameter between previously given ranges. Optimal parameter values are as follows according to this data set; population size 152.727, reference set ratio 70%, good reference set ratio %66.67, diverse reference set ratio 33.33%. These values are produced by optimizing response of the regression model of the test data. Optimal value for population size results as 152.727 however can be safely rounded down to 150 to also enable subsets to be in integers while keeping the change minimal. Likewise, good and diverse reference set ratios are also rounded to the nearest integer. Effect of difference in iteration number on makespan is still not statistically significant. As a result, 20 is considered as an acceptable value for iteration number. If time is not a concern it is suggested to increase iteration number or run the algorithm more than once. In conclusion optimal parameters can be rounded and summarized as in Table 2.

Tab. 2. Optimal parameter values

Parameter	**Optimal Values**
Iteration	20
Population size	150
Reference set size	105 (70%)
Good reference set size	70 (67%)
Diverse reference set size	35 (33%)

Tab. 3. Results produced using optimal parameters

Problem	**Riahi's Parameters**		**Optimal Parameters**		**Problem**	**Riahi's Parameters**		**Optimal Parameters**	
	Cmax	**Runtime (ms)**	**Cmax**	**Runtime (ms)**		**Cmax**	**Runtime (ms)**	**Cmax**	**Runtime (ms)**
ta001	1286	72	1286	919	ta031	2729	410	2729	14092
ta002	1365	112	**1359**	932	ta032	2838	260	2838	2734
ta003	1116	88	**1098**	884	ta033	2633	388	**2629**	3838
ta004	1320	78	**1317**	673	ta041	3118	600	**3109**	9028
ta005	1305	103	**1264**	2298	ta042	3020	478	3020	5476
ta006	1224	85	**1210**	1849	ta043	2960	493	2960	5769
ta007	1251	61	1251	1561	ta051	4023	1104	**4021**	16108
ta008	1211	79	1211	1587	ta052	**3900**	1148	3902	13335
ta009	1266	53	**1253**	1251	ta053	3912	2018	**3870**	15988
ta010	1133	65	**1108**	833	ta061	5519	2445	**5512**	53166
ta011	1625	119	**1616**	1778	ta062	5284	1064	5284	32265
ta012	1729	143	**1692**	1339	ta063	5206	919	5206	14857
ta013	1511	156	**1508**	1725	ta071	5826	1862	5826	28398
ta014	1411	102	1411	1490	ta072	5400	2086	5400	39226
ta015	**1478**	95	1493	1366	ta073	5785	1830	**5757**	26929
ta016	1433	99	**1425**	1233	ta081	**6473**	3981	6495	51882
ta017	1526	151	1526	3972	ta082	6516	3854	**6480**	54263
ta018	1600	123	**1579**	1209	ta083	**6584**	4427	6590	72963
ta019	1614	137	**1612**	1412	ta091	10942	13888	10942	90288
ta020	1634	109	1634	1532	ta092	10706	7479	**10686**	85327
ta021	2329	266	**2310**	3929	ta093	11025	7745	11025	89231
ta022	2150	232	**2142**	3007	ta101	11563	15097	**11550**	178674
ta023	2357	249	**2354**	3898	ta102	11641	14261	**11625**	202037
					ta103	11814	15068	**11784**	212474

A data set is created using the optimal parameters found and compared to a second data set produced by using recommended parameters proposed by Riahi et al., 2017. Overall, an improvement is made to makespan on 27 of 47 problems, 16 reached the same value, 4 had worse values. A comparison is given in Table 3.

On the other hand, average algorithm runtime is increased 15 times mainly due to the large population size. Relation between algorithm runtime in milliseconds and the number of machines (m) and jobs (j) is formulated as follows:

$$Runtime\ (ms) = 12358 - 272\,m - 1780\,j + 1{,}261\,m^2 + 53.9\,m * j$$

A comparison between the best-known solutions of Taillard's benchmark problems (Taillard, 1993) and the algorithm results acquired with the newly found optimal parameters are given in Table 4. SSA represents the best results gathered by the Scatter Search Algorithm of this study while upper bound is the best-known results of the respective problem. Normalized difference percent is also given in the dif column. On average, results differ by 2.21 percent from the best-known solutions. Out of 47 problems tested, 2 results were able to match the best-known solution while the worst result was only 6.32% apart makespan wise. Increase in the number of machines and jobs in problem models does not appear to have an effect on algorithm's ability to produce similar results to best-known solutions.

Table 4 also includes best solutions from NEH (Nawaz et al., 1983), Social Spider Optimization (SSO) (Kurdi, 2021), A Hybrid Whale Optimization Algorithm (HWA) (Abdel-Basset et. al., 2018) and Refining Decomposition-based Integrated Search (RDIS) (Amirghasemi, 2021) as a comparison. While the algorithm in study produced better results than NEH and similar results to SSO, HWA and RDIS fairly outperforms the rest.

Tab. 4. Results compared to best known values

Problem	Upper bound	Dif (%)	SSA	NEH	SSO	HWA	RDIS
ta001	1278	0,63	1286	1286	1282	1278	1278
ta002	1359	0	1359	1365	1359	1359	1359
ta003	1081	1,57	1098	1159	1088	1081	1081
ta004	1293	1,86	1317	1325	1300	1293	1293
ta005	1235	2,35	1264	1305	1237	1235	1235
ta006	1195	1,26	1210	1228	1195	1195	1195
ta007	1234	1,38	1251	1278	1243	1239	1239
ta008	1206	0,41	1211	1223	1206	1206	1206
ta009	1230	1,87	1253	1291	1231	1230	1230
ta010	1108	0	1108	1151	1108	1108	1108
ta011	1582	2,15	1616	1680	1598	1582	1582
ta012	1659	1,99	1692	1729	1682	1659	1659
ta013	1496	0,8	1508	1557	1513	1496	1496
ta014	1377	2,47	1411	1439	1395	1377	1377
ta015	1419	5,21	1493	1502	1440	1419	1419
ta016	1397	2	1425	1453	1404	1397	1397
ta017	1484	2,83	1526	1562	1493	1484	1484
ta018	1538	2,67	1579	1609	1555	1538	1538
ta019	1593	1,19	1612	1647	1606	1593	1593
ta020	1591	2,7	1634	1653	1611	1591	1591
ta021	2297	0,57	2310	2410	2329	2297	2297
ta022	2099	2,05	2142	2150	2125	2099	2099
ta023	2326	1,2	2354	2411	2350	2326	2326
ta031	2724	0,18	2729	2733	2724	2724	2724
ta032	2834	0,14	2838	2843	2839	2834	2836
ta033	2621	0,31	2629	2640	2621	2621	2621
ta041	2991	3,95	3109	3135	3053	3021	3025
ta042	2867	5,34	3020	3032	2938	2891	2911
ta043	2839	4,26	2960	2986	2890	2869	2871
ta051	3850	4,44	4021	4082	3974	3876	3917
ta052	3704	5,35	3902	3921	3808	3715	3757
ta053	3640	6,32	3870	3927	3772	3653	3699

Tab. 4. (continued)

Problem	Upper bound	Dif (%)	SSA	NEH	SSO	HWA	RDIS
ta061	5493	0,35	5512	5519	5493	5493	5493
ta062	5268	0,3	5284	5348	5284	5268	5268
ta063	5175	0,6	5206	5219	5193	5175	5175
ta071	5770	0,97	5826	5846	5787	5776	5779
ta072	5349	0,95	5400	5453	5379	5362	5353
ta073	5676	1,43	5757	5824	5691	5691	5679
ta081	6202	4,72	6495	6541	6377	6280	6369
ta082	6183	4,8	6480	6523	6360	6278	6303
ta083	6271	5,09	6590	6639	6450	6368	6385
ta091	10862	0,74	10942	10942	10947	10885	10885
ta092	10480	1,97	10686	10716	10542	10512	10503
ta093	10922	0,94	11025	11025	11005	10965	10965
ta101	11195	3,17	11550	11594	11418	11335	11399
ta102	11203	3,77	11625	11675	11488	11517	11482
ta103	11281	4,46	11784	11852	11559	11481	11535

4. Conclusion

The PFSP is one of the job shop scheduling problems. PFSP is known as an NP-hard problem from the literature. In this study, scatter search method is proposed to compare the PFSP problems. The proposed method is iterated the local and global search method for the initial population. In this study, it is aimed to optimize the PFSP with the scatter search method. The flow scheduling problem in the literature was considered as test problems and the objective function was determined as the minimization of the total completion time of the jobs. Unlike the studies in the literature, the scatter search model was analyzed with the algorithm prepared with the JavaScript programming language. Best solutions acquired by the algorithm in subject are found within 2% in range on average compared to the best-known solutions. Variation between these solutions is correlated to neither problem size nor the algorithm parameters but to the complexity in processing times of the jobs. Proposed parameters, especially the population size is the main reason for the increased runtime but on a modern computer with more processing power, the runtimes would decrease. In addition, the solutions can be improved by parameter optimization with different experimental design methods. As scatter search heavily depends on exploration via good and diverse solutions while the good solutions being the primary pulling power due to having better makespan, it is recommended to enrich the initial population with more starter algorithms in the diversification method.

References

1. **Abdel-Basset, M., Manogoran, 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: 129-145.
2. **Abdelmaguid, T.F. (2020).** Scatter search with path relinking for multiprocessor open shop scheduling. *Computers & Industrial Engineering*, 141, 1-19.
3. **Abdollahzadeh, B., Soleimanian Gharehchopogh, F., & Mirjalili, S. (2021**). Artificial gorilla troops optimizer: a new nature-inspired metaheuristic algorithm for global optimization problems. *International Journal of Intelligent Systems*, 36(10), 5887-5958.
4. **Alharkan, M.I. (2005).** Algorithms for Sequencing and Scheduling, King Saud University, Riyadh.
5. **Amirghasemi, M. (2021).** An Effective Decomposition-Based Stochastic Algorithm for Solving the Permutation Flow-Shop Scheduling Problem. *Algorithms*, 14, 112.
6. **Arshad, A., Gajpal, Y. & Elmekkawy, T.Y. (2021).** Distributed permutation flowshop scheduling problem with total completion time objective. *Opsearch*, 58(2), 425-447.
7. **Baskar, A. & Xavior, M. A. (2021).** New idle time-based tie-breaking rules in heuristics for the permutation flowshop scheduling problems. *Computers & Operations Research*, 133, 105348.
8. **Behnamian, J., Memar Dezfooli, S., & Asgari, H. (2021).** A scatter search algorithm with a novel solution representation for flexible open shop scheduling: a multi-objective optimization. *The Journal of Supercomputing*, 77(11), 13115-13138.

9. **Campbell, H. G., Dudek, R. A. & Smith, M. L. (1970).** A heuristic algorithm for the n job, m machine sequencing problem. *Management science*, 16(10), B-630.
10. **Çiçekli, U.G. & Bozkurt S. (2016).** Permütasyon akış tipi çizelgeleme probleminin dağınık arama ile optimizasyonu. *Ege Akademik Bakış*, 16, 31-40.
11. **Erol, V. (2006).** *Design and implementation of a population and neighborhood-based metaheuristic algorithm for vehicle routing problems*, Master Thesis, Yıldız Technical University, İstanbul, Turkey.
12. **Fink, A. & Voß, S. (2003).** Solving the Continuous Flow-Shop Scheduling Problem by Metaheuristics. *European Journal of Operational Research*, 151: 400-414.
13. **Graves, S.C. (1981).** A Review of Production Scheduling. *Operations Research*, 29(4): 646-675.
14. **Kaya S. & Fığlalı N. (2018**). Use of meta-heuristic methods to solve the multi-objective flexible job shop scheduling problems, *Harran University Journal of Engineering*, 3(3), 222-233.
15. **Kaya, S., Aydilek, İ.B., Tenekeci M.E. & Gümüşçü A. (2020).** The effects of initial populations in the solution of flow shop scheduling problems by hybrid firefly and particle swarm optimization algorithms. *Pamukkale University Journal of Engineering Sciences*, 26(1), 140-149.
16. **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.
17. **Külahlı, S., Engin, O., & Koç, İ. (2021).** A New Hybrid Scatter Search Method for Solving the Flexible Job Shop Scheduling Problems. *Celal Bayar University Journal of Science*, 17(4).
18. **Marti, R., Laguna, M. & Glover, F. (2006**). Principles of Scatter Search, *European Journal of Operational Research*, 169:359-372.
19. **Mashuri C., Mujianto A.H., Sucipto H., Arsam R. Y. & Permadi G.S. (2019).** Production Time Optimization using Campbell Dudek Smith (CDS) Algorithm for Production Scheduling, *E3S Web of Conferences* 125, 23009.
20. **Mete, U. (2019).** *A variable neighborhood search approach for permutation flowshop*. Scheduling. Master's Thesis, Pamukkale University, Turkey.
21. **Moghaddam, R.T., Javadian, N., Khorrami, A. & Gholipour-Kanani Y. (2010).** Design of a scatter search method for a novel multi-criteria group scheduling problem in a cellular manufacturing system, *Expert Systems with Applications*, 37 ,2661–2669.
22. **Nawaz, M., Enscore Jr, E. E. & Ham, I. (1983).** A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega*, 11(1), 91-95.
23. **Nowicki E. & Smutnicki C., (2006).** Some aspects of scatter search in the flow-shop problem, *European Journal of Operational Research*, 169, 654–666.
24. **Oktay, S. & Engin, O., (2006).** Scatter search method for solving industrial problems: literature survey. *Journal of Engineering and Natural Sciences*, 3, 144-155.
25. **Osman, I.H. & Laporte, G. (1996).** Metaheuristics: a bibliography. *Annals of Operations Research*, 63, 513-623.
26. **Osman, I.H. & Kelly, J.P. (1996).** Meta-heuristics: an overview. *Meta-heuristics*, 1-21.
27. **Palmer, D.S. (1965).** Sequencing jobs through a multi-stage process in the minimum total time - a quick method of obtaining a near optimum. *Journal of the Operational Research Society*, 16(1), 101-107.
28. **Pan, Q.K., Gao, L., Wang, L., Liang, J. & Li, X.Y. (2019).** Effective heuristics and metaheuristics to minimize total flowtime for the distributed permutation flowshop problem. *Expert Systems with Applications*, 124, 309-324.
29. **Rahimi-Vahed, A.R., Javadi, B., Rabbani, M. & Moghaddam, R.T. (2008).** A multi-objective scatter search for a bi-criteria nowait flow shop scheduling problem, *Engineering Optimization*, 331-346.
30. **Riahi V., Khorramizade M., Hakim Newton M.A. & Sattar A. (2017).** Scatter search for mixed blocking flowshop scheduling, *Expert Systems with Applications* 79:20-32.
31. **Rimli M.A., Deris S., Mohamad M.S., Omatu S. & Corchado J.M. (2017).** An enhanced scatter search with combined opposition-based learning parameter estimation in large-scale kinetic models of biochemical systems, *Engineering Application of Artificial Intelligence* 62, 164-180.
32. **Sadiq, A. & Muhamad, K. (2012).** Improved scatter search for job shop scheduling problem. *International Journal of Research and Reviews in Soft and Intelligent Computing*, 2(1), 104-107.
33. **Sagarna, R. & Lozano, J. A. (2006).** Scatter Search in Software Testing, Comparison and Collaboration with Estimation of Distribution Algorithms, *European Journal of Operational Research*, 169(2):392-412.
34. **Saravanan M. & Haq A.N. (2008).** Evaluation of Scatter Search Approach for Scheduling Optimization of Flexible Manufacturing Systems, *The International Journal of Advanced Manufacturing Technology*, 38, 978–986.
35. **Stützle T. & Hoos H.H. (2000).** MAX-MIN Ant System, *Future Generation Computer Systems* 16(8):889-914.

36. **Taillard E., (1993).** Benchmarks for basic scheduling problems, *European Journal of Operational Research*, 64(2): 278-285.
37. **Yang Y., Li P., Wang S., Liu B. & Luo Y. (2017).** Scatter Search for Distributed Assembly Flowshop Scheduling to Minimize Total Tardiness, *IEEE*, 861-868.