EasyChair Preprint

№ 8962

Monocriterion Optimization in Parallel Flow Shop.

Milad Mansouri, Younes Bahmani and Hacene Smadi

October 3, 2022

Monocriterion optimization in Parallel Flow Shop.

Milad Mansouri[1], Younes Bahmani[1],
and Hacene Smadi[1]

[1]Laboratory of Automation and Manufacturing, University Batna2, street Chahid Boukhlouf, Batna, 05000, Algeria
{Milad Mansouri, E-mail: milad.mansouri@univ-batna2.dz
Younes Bahmani, E-mail: y.bahmani@ univ-batna2.dz
Hacene Smadi, E-Mail: h.smadi@univ-batna2.dz}

Abstract. Scheduling problems have always been widely discussed in the literature, such as the classical flow-shop scheduling problem (single path shop). However, the Parallel Flow-Shop Scheduling Problem (PFSSP), has received very little attention due to its complexity to be studied. In this paper, we focus on the latter, by proposing two nature-inspired metaheuristics (Genetic algorithm and Particle Swarm Optimization), to minimize the criterion Makespan Finally, a comparison of results was proposed using a performance index.

Keywords: Parallel Flow shop Scheduling Problem, Makespan, Metaheuristics, Monocriterion optimization.

1 Introduction

In this paper, we propose two approaches to optimize the scheduling function in order to improve productivity within an industry, we find various layouts of machines and lines within a company, called type of workshops, each industry includes a type of workshop corresponds to it. There are mainly three types of workshops: flow-shop, job-shop and open-shop and, each of these three workshops has in turn secondary types; for example: generalized open shop, flexible job-shop, hybrid flow-shop and many others. According to the literature, the scheduling of the workshops mentioned before have been widely studied and continue to be discussed, unlike the parallel flow-shop which has received very little attention and this comes down to the fact that it is composed of two sub-problems: assignment of jobs to production lines and planning of the order of passage of assigned jobs. The fact that this type is little approached has motivated us to take advantage of it.

A PFSSP is an extension of a classic flow shop, in other words; mind it is the arrangement of several identical lines (flow-shop) in a parallel and independent way. keeping this in mind, we are carrying out an experiment to minimize the Makespan objective function by implementing two metaheuristics: genetic algorithm and particle swarm. This choice of metaheuristics is due to the combinatorial problem of a complexity NP-hard. In this work, we have hypothesized that there is no waiting time between two adjacent machines, this means that when a job i finishes being processed on machine j it will directly join the next machine j+1.

Indeed, according to the experiment to carry out the GA (Genetic Algorithm) and PSO (Particle Swarm Optimization) give very close results, however for the problems of small size the method PSO gives better results than the GA but for the problems of large size it is the GA who excels.

2 LITERATURE REVIEW OF PFSSP PROBLEM

There are very few works in the literature that deal with the parallel flow shop scheduling problem, this field is still virgin compared to other types such as: the classic flow-shop or flexible flow-shop, the most important works on the subject are mentioned in this section.

In 1992, [1] evaluated the performance of algorithms like SPT applied already on classical flow shop on the two-stage parallel flow-shop and the author compared the result found with the branch-and-bound heuristic seeking to minimize total flow time. Computational experience has shown that the branch-and-bound heuristic gives satisfying results in terms of solution quality and, the computation time is significantly better than the SPT rule already applied on old classic flow shop jobs. Then in 2003 [2], proposed to minimize a weighted sum of production cost and the cost incurred from late product delivery in a m parallel flow-shop, each one has 02 machines, he employed a heuristic algorithm combining Tabu search and Johnson's method. A year later [3], did a research to minimize the makespan by a multi-phase heuristic algorithm in a workshop of two parallel flow shops with proportional processing times and he also proposed a simulation study to evaluate the effectiveness of his proposed heuristic for small and large size problems. In 2011, [4] in order to minimize makespan in m two-stage parallel flow shops; k sequential machine by flow, authors proposed to divide the general problem into two sub-problems, the first consists in assigning the jobs to parallel flow workshops and the second consists in planning these tasks to be assigned using Johnson's rule . And an approximation algorithm is proposed for the case of m=2 and m=3. In the same year, [5] proposes a mathematical model for the parallel flow-shop problem in order to minimize the makespan by a Quantum algorithm.

Recently, in 2017 [6] and in order to minimize makespan in a workshop of F identical parallel flow shops and a constraint of blocking of machines (this type of constraint for this type of problem has never been studied according to authors), and for that they chose constructive and improvement heuristics but by two different approaches than those often proposed with the aim of minimizing the maximum processing end time between lines. Then in 2018, [7] proposed an iterated greedy algorithm to solve the same type of problem, already mentioned [6], but this time for another objective function which is: minimization of the total tardiness of jobs. And lately in 2021[8], in order to minimize makespan for a parallel flow shop problem with blockage of machines and with sequence-dependent setup times, they choosed to combine 35 sequencing rules with allocation methods, then tested a heuristic (RCP0), which takes a different approach that is specially adapted to the problem under study, The result of the experiment carried out to validate the heuristic (RCP0), approved its efficiency and performance.

3 PFSSP DESCRIPTION

Parallel Flow-shop (workshop with multiple single paths) type of shop was created to cover the inability of a classic flow-shop to produce enough product in a reasonable time.
A classic flow-shop (single-path workshop):is a production line which consists of *m* machines in series, and the order of passage on the various machines is the same for all the jobs (i.e., a job cannot leave the line before traversing all the machines from M_1 to M_m).

A parallel flow shop is the superposition of identical lines of a classic flow-shop placed in a parallel and independent way.

Description of parallel flow-shop scheduling problem:

a. We have N products (or jobs) which will all have to be affected and then processed by three 03 lines in our case.
b. Each product can only be processed in one production line, and once assigned to a line, the product will not be able to leave it and must cross all m machines (from M_1 to M_m).
c. A line can manufacture several products.
d. Each line is in turn composed by *m* machines serially ordered.
e. Each machine can run at most one product at a time; each product can be processed on at most one machine at a time.
f. Products are processed without interruption.
g. Processing times $P_{i,j}$ of products, are known and set beforehand.

4 METAHEURISTICS

Before choosing a resolution method, we should first study the complexity of the problem.
The PFSSP is an NP-hard combinatorial problem and that's why we propose to solve it with metaheuristic methods.
In our case, we have chosen the genetic algorithm and the particle swarm method.

4.1 Genetic algorithm

The proposed genetic algorithm is an evolutionary metaheuristic algorithm which has a group of feasible solutions, it is designed for complex problems that cannot be solved by exact methods. The genetic algorithm has proven itself for the resolution of combinatorial optimization problems, the intensive use of this method in solving NP-hard combinatorial problems also motivated us to use it in our problem and compared it with another method (particle swarm optimization) to judge its effectiveness.
In 1975 JH Holland, proposed the concept of genetic algorithms and since then several considerable researches have adopted and attempted to improve this method such as [9], [10],[11] and many others. Inspiring their names and processes from biological evolution of species. Their unfolding process is composed of three stages: generation of the initial population, evaluation mechanism and evolutionary mechanism.
A solution is a succession of jobs, the solutions are called chromosomes or individuals, each chromosome is a sequence of a finite number of genes where each gene in turn represents a job. For each chromosome, we calculate the fitness function, the idea is to keep for each iteration (generation) a certain number of efficient solutions which have the smallest fitness value for the reproduction of future generations. Children are

generated using a crossover operator applied to parents, some individuals are modified (mutations) by a mutation operator. This process is repeated as long as the stopping condition is not satisfied.

4.2 Particle swarm optimization

Introduced by [12], this metaheuristic is originally intended for continuous optimization problems, recently several attempts have been made to apply it to discrete problems. The solution is represented by a particle, and the particle is characterized by a position, a memory and a velocity in the search space. The particle's memory is used to store the best position reached by the particle and also the best position visited by all the particles that make up the swarm. This algorithm is also called iterative because at each iteration the particles move according to a speed in the search space. Its stopping criterion is generally specified by the number of generations.

5 COMPUTATIONAL EXPERIMENTS

In this section, we will describe the experiment carried out and discuss the results.

5.1 Experience

In order to carry out our experiment; we have programmed two solution population metaheuristics (GA-PSO) by the MATLAB programming environment; and the number of parallel lines is fixed at 03.

The steps of the experiment are detailed as follows:

a. The experiment was carried out on 12 benchmarks (each of these benchmarks is composed of 10 different instances) and 20 trials were established on each of these twelve benchmarks.
b. Each benchmark has two elements: number of products to be processed and number of processing machines.
   The values of the benchmark elements are different from each other.
c. For small size problem, we considered the following dimensions (20 5; 20 10;20 20;50 05;50 10; 50 20).
d. For large size problem, we took the following dimensions (100 05;100 10;100 20;200 10;200 20;500 20).
e. Each trial gives us the following results: C max of the 1st line, Cmax of the 2nd line, C max of the 3rd line, their average and total execution time.
f. The first 10 Makespan values out of 20 were selected after sorting them in ascending order.
g. For a better interpretation of the results, we have calculated the relative index of deviation (RDI):

$$\mathrm{RDI} = \frac{\mathrm{Alg}_{\mathrm{sol}} - \min_{\mathrm{sol}}}{\max_{\mathrm{sol}} - \min_{\mathrm{sol}}} \times 100 \qquad \textbf{(1)}$$

The values given by this index vary between [0-100], the closer the RDI value is to zero, the better it is,[13]**.** It's used to make better decision when the results obtained from different methods are very close.
A summary of the results of the experiment described above, is shown in Table1.

Table1. The results of the experiment for the 12 instances.

	GA						**PSO**					
	Line 1	**Line 2**	**Line 3**	**Cmax**	**Time [s]**	**RDI_GA [%]**	**Line 1**	**Line 2**	**Line 3**	**Cmax**	**Time [s]**	**RDI_PSO [%]**
20_05	615	632	631	632	0,2123	18,07	560	623	615	623	0,2003	16,21
20_10	887	911	915	915	0,3161	34,32	905	890	806	905	0,3048	28,50
20_20	1530	1542	1524	1542	0,5034	39,18	1486	1507	1531	1531	0,5493	38,69
50_05	1126	1206	1212	1212	0,4735	21,15	1171	1104	1170	1171	0,4907	24,91
50_10	1576	1623	1614	1623	0,7219	33,32	1566	1587	1576	1587	0,8672	50,59
50_20	2402	2400	2225	2402	1,2710	50,33	2273	2285	2323	2323	1,4155	66,52
100_05	2131	2002	2091	2131	0,9117	42,90	2141	2074	2094	2141	1,1744	103,28
100_10	2570	2609	2595	2609	1,4234	33,61	2632	2588	2572	2632	1,7135	95,29
100_20	3476	3530	3489	3530	2,5680	38,44	3488	3417	3491	3491	2,8160	106,06
200_10	4530	4616	4612	4616	3,0649	52,47	4564	4541	4412	4564	3,3485	106,46
200_20	5582	5604	5560	5604	4,8081	56,43	5504	5616	5598	5616	5,3606	152,85
500_20	11517	11516	11472	11517	12,0729	94,81	11511	11416	11387	11511	13,4241	157,05

For better visibility, we have divided the instances into two classes (**Fig. 1** represents the values of Cmax according to small instances), (**Fig. 2** represents the values of Cmax according to large instances).

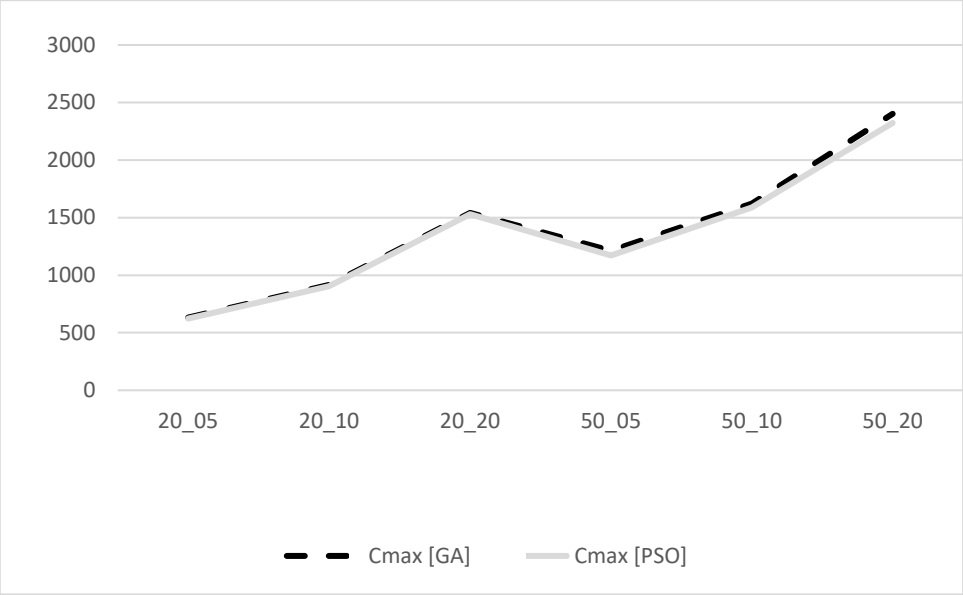

Fig. 1. Represents the values of Cmax of GA and PSO for small instances. The abscise axis represents the small instances and the ordinate axis represents the values of Cmax for GA and PSO. The resulting curves show that the two heuristics give almost similar results with a sligh difference in favor of PSO for the small instances.

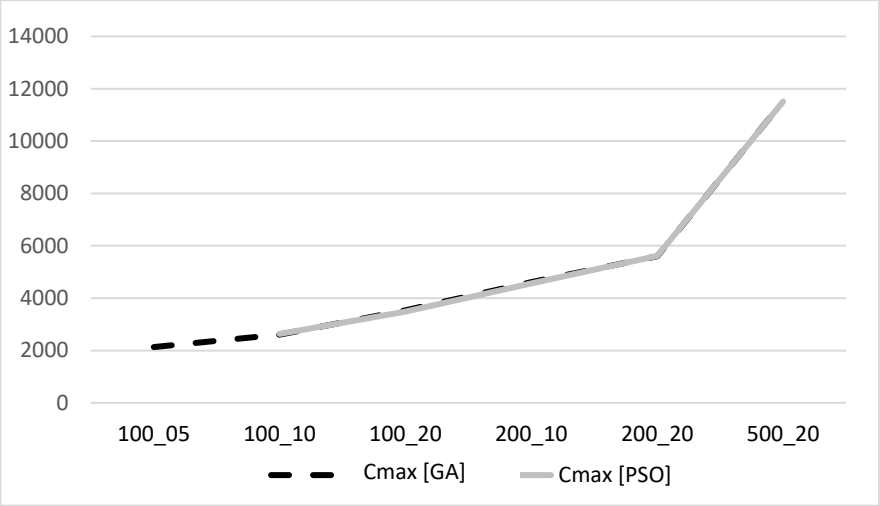

Fig. 2. Represents the values of Cmax of GA and PSO for large instances. The abscise axis represents the large instances and the ordinate axis represents the values of Cmax for GA and

PSO. The variation of the curves is almost similar, but in this case; genetic algorithm gives better results than particle swarm.

Fig. 3. shows a comparison of computation times for the two heuristics according to all the instances

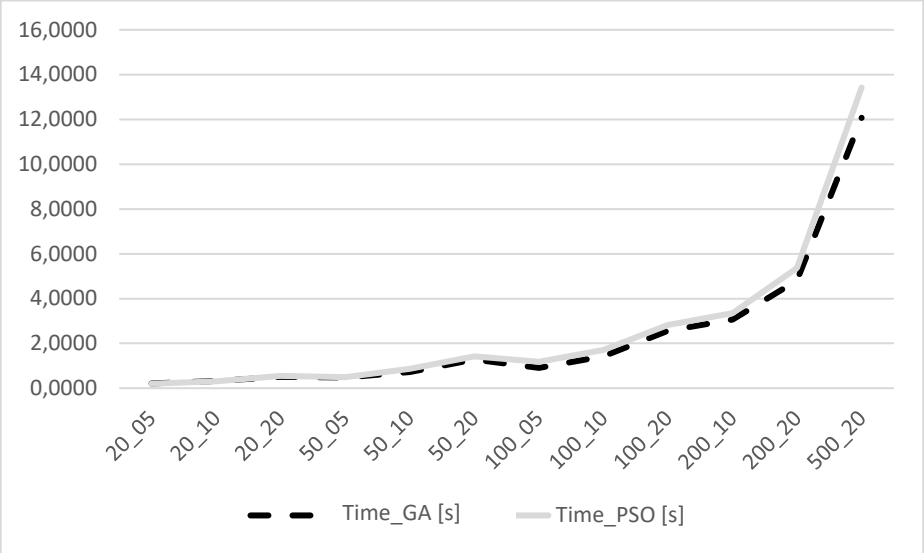

Fig. 3. Represents the computation time values of GA and PSO. The abscise axis represents the instances and the ordinate axis represents the computation time values of GA and PSO. The computation time increases in both cases, however the execution time for GA increases progressively; contrary to the PSO, the increase in computation time is a bit faster.

According to this interpretation, overall neither of the two algorithms is more efficient than the other for the case of PFSSP and it is for this reason that we have opted for the calculation of RDI in order to be able to decide which of the two heuristics is better.

The overall RDI of GA is: 42.92% and of PSO is: 78.87%.

This allows us to say that, for 03 parallel flow-shop scheduling problem; the GA perform better than PSO.

6 CONCLUSION

In this paper we have accentuated our study on parallel flow-shop scheduling problem [14],[15],[16]; by the implementation of two metaheuristic algorithms (GA-PSO). In the experiment that we carried out, we have considered this type of FPSSP workshop

with three parallel lines, and as a criterion, we have chosen to minimize the objective function Makespan which is used to minimize the total duration of the schedule, in order to establish a sequence of jobs for each line with the minimum processing end time possible, and as a constraint we have considered that there is no buffer zone between two adjacent machines. In other words, there is no waiting time for a job between two machines. The results found by the two algorithms are very encouraging. Although the RDI index allowed us to say that GA is better than PSO for the parallel flow-shop scheduling problem.

References

[1] C. Rajendran, D. Chaudhuri, Theory and Methodology A multi-stage parallel-processor flowshop problem with minimum flowtime, 57 (1992) 111–122.

[2] P. Taylor, D. Cao, M. Chen, International Journal of Production Parallel flowshop scheduling using Tabu search, (2010) 37–41. https://doi.org/10.1080/0020754031000106443.

[3] A. Al-salem, A Heuristic to Minimize Makespan in Proportional Parallel Flow Shops, 2 (2004) 98–107.

[4] X. Zhang, S. Van De Velde, Approximation Algorithms for the Parallel Flow Shop Problem, Eur. J. Oper. Res. (2011). https://doi.org/10.1016/j.ejor.2011.08.007.

[5] Y. Jiang, S. Wan, Parallel Flow Shop Scheduling Problem Using Quantum Algorithm, (2011) 269–274.

[6] I. Ribas, R. Companys, X. Tort-martorell, US CR IT AC PT, (2017). https://doi.org/10.1016/j.eswa.2017.01.006.

[7] I. Ribas, R. Companys, X. Tort-martorell, US CR IT AC PT, Expert Syst. Appl. (2018). https://doi.org/10.1016/j.eswa.2018.12.039.

[8] I. Ribas, R. Companys, International Journal of Industrial Engineering Computations A computational evaluation of constructive heuristics for the parallel blocking flow shop problem with sequence-dependent setup times, 12 (2021) 321–328. https://doi.org/10.5267/j.ijiec.2021.1.004.

[9] S.K. Iyer, B. Saxena, Improved genetic algorithm for the permutation owshop scheduling problem, 31 (2004) 593–606. https://doi.org/10.1016/S0305-0548(03)00016-9.

[10] R. Ruiz, J.C. García-díaz, C. Maroto, Considering scheduling and preventive maintenance in the flowshop sequencing problem, 34 (2007) 3314–3330. https://doi.org/10.1016/j.cor.2005.12.007.

[11] I.H. Dridi, R. Kammarti, M. Ksouri, P. Borne, I.H. Dridi, R. Kammarti, M. Ksouri, P. Borne, G. Algorithm, Genetic Algorithm for Mulicriteria Optimization of a Multi-Pickup and Delivery Problem with Time Windows To cite this version : HAL Id : hal-00521535 Genetic Algorithm for Mulicriteria Optimization of a Multi-Pickup and Delivery Problem with Time Windows, (2010).

[12] J. Kennedy, R. Eberhart, Particle Swarm Optimization, (1995) 1942–1948.

[13] E. Mehdizadeh, R. Tavakkoli-Moghaddam, M. Yazdani, A vibration damping optimization algorithm for a parallel machines scheduling problem with sequence-independent family setup times, Appl. Math. Model. 39 (2015) 6845–6859. https://doi.org/10.1016/j.apm.2015.02.027.

[14] ICIST '20: Proceedings of the 10th International Conference on Information Systems and Technologies, isbn: 9781450376556, Lecce, Italy. ACM, 2020.

[15] Preface Conference Proceedings: 2021 International Conference on Information Systems and Advanced Technologies (ICISAT). Mohamed Ridda Laouar, IEEE, 2021.
[16] Effects of Interaction on E-Learning Satisfaction and Outcome: A Review of Empirical Research and Future Research Direction. Mohamed Ridda LAOUAR and Sean B. Eom. International Journal of Information Systems and Social Change, 2017.