Improving NSGA-III for flexible job shop scheduling using automatic configuration, smart initialization and local search

Wang, Y.; Stein, N. van; Bäck, T.H.W.; Emmerich, M.T.M.

Improving NSGA-III for Flexible Job Shop Scheduling using Automatic Configuration, Smart Initialization and Local Search

Yali Wang
Bas van Stein
y.wang@liacs.leidenuniv.nl
b.van.stein@liacs.leidenuniv.nl
Leiden University
Leiden, The Netherlands

Thomas Bäck
Michael Emmerich
t.h.w.baeck@liacs.leidenuniv.nl
m.t.m.emmerich@liacs.leidenuniv.nl
Leiden University
Leiden, The Netherlands

ABSTRACT

This paper provides a short summary of a novel algorithm tailored towards multi-objective flexible job shop scheduling problems (FJSP). The result shows that for challenging real-world problems in combinatorial optimization, off-the-shelf implementations of multi-objective optimization evolutionary algorithms (MOEAs) might not work, but by using various adaptations, these methods can be tailored to provide excellent results. This is demonstrated for a state of the art MOEA, that is NSGA-III, and the following adaptations: (1) initialization approaches to enrich the first-generation population, (2) various crossover operators to create a better diversity of offspring, (3) parameter tuning, to determine the optimal mutation probabilities, using the *MIP-EGO configurator*, (4) local search strategies to explore the neighborhood for better solutions. Using these measures, NSGA-III has been enabled to solve benchmark multi-objective FJSPs and experimental results show excellent performance.

CCS CONCEPTS

• **Computing methodologies → Planning and scheduling**; • **Applied computing → Multi-criterion optimization and decision-making**;

KEYWORDS

Flexible job shop scheduling, multi-objective optimization, evolutionary algorithm, algorithm engineering

ACM Reference Format:
Yali Wang, Bas van Stein, Thomas Bäck, and Michael Emmerich. 2020. Improving NSGA-III for Flexible Job Shop Scheduling using Automatic Configuration, Smart Initialization and Local Search. In *Genetic and Evolutionary Computation Conference Companion (GECCO '20 Companion), July 8–12, 2020, Cancún, Mexico.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3377929.3389924

GECCO '20 Companion, July 8–12, 2020, Cancún, Mexico

ACM ISBN 978-1-4503-7127-8/20/07.
https://doi.org/10.1145/3377929.3389924

1 INTRODUCTION

The flexible job shop scheduling problem (FJSP) is an important extension of the classical job shop scheduling problem which allows an operation to be processed by more than one machine. The problem is not only to assign each operation to a machine out of a set of qualified machines, but also to order the operations on the machines. In this paper, evolutionary algorithm (EA) has been applied to the multi-objective flexible job shop scheduling problem (MOFJSP) with three conflicting objectives, namely: the makespan (i.e., the maximal completion time of all operations), total workload and critical workload (i.e., the maximum workload among all machines). We adopt multiple initialization approaches to enrich the first-generation population; at the same time, diverse genetic operators are applied to guide the search towards offspring with a wide diversity; especially, we use an algorithm configurator to tune the parameter configuration; furthermore, local search is employed leading to better solutions. In general, the proposed EA can be combined with any multi-objective evolutionary algorithm (MOEA), we pick NSGA-III to investigate its performance because of its state-of-the-art performance in multi-objective optimization.

2 PROPOSED EVOLUTIONARY ALGORITHM

The algorithm follows a typical EA procedure, augmented by local search. Each individual is encoded as a chromosome consisting of two parts: (1) The **operation sequence vector** consists of only job indexes. For each job, the first appearance of its index represents the first operation of that job and the second appearance of the same index represents the second operation of that job, and so on. (2) The **machine assignment vector** presents the selected machines for all operations. We consider different approaches for initializing the machine assignment and operation sequence vectors. E.g., when generating the machine assignment vector, assign a random machine, assign the machine with the most number of operations, and so on; when generating the operation sequence vector, the operations of jobs are shuffled, and so on. In our algorithm, several new methods are proposed and adopted together with a few classical known methods to provide enough population diversity[1]. When generating a new individual, one initialization method for the machine assignment vector and one for the operation sequence vector are picked randomly to enrich the initial population. Since our representation of chromosomes has two parts, crossover and mutation operators applied to these two parts of chromosomes are

[1]For a complete list of operators and further details, see [4].

implemented separately. Similarly, several new crossover operations are proposed and used together with several other operators from literature. In our algorithm, one crossover operator for machine assignment and one operator for operation sequence are randomly chosen to generate the offspring. Furthermore, insertion mutation and swap mutation (one- and two-point swap) are used.

Decoding a chromosome is to convert an individual into a feasible schedule to calculate the objective values which represent the relative superiority of a solution. In this process, the operations are picked one by one from the operation sequence vector and placed on the machines from the machine assignment vector to form the schedule. When placing each operation to its machine, local search is involved to refine an individual in order to obtain an improved schedule. Two levels of local search are applied in our algorithm to reallocate operations to better time slots which are idle times exist between operations on each machine due to precedence constraints among operations of each job. The first level local search checks the idle time intervals for each operation on its current machine and switches the operation to early idle time when possible. The second level local search checks the idle time intervals for each operation on its alternative machines and switches the operation to early idle time when possible.

3 EXPERIMENTS AND RESULTS

The algorithms are tested on two sets of well-known benchmark instances: 4 Kacem instances (ka4x5, ka10x7, ka10x10, ka15x10) and 10 BRdata instances (Mk01-Mk10). All the experiments are performed with a population size of 100, and the evaluation budget of 10, 000 for Kacem instances and 150, 000 for BRdata instances.

The crossover probability is set to 1 and two random crossover operators can be chosen each time (one for operation sequence and one for machine assignment). For Kacem instances, the mutation probabilities are set to 0.6. For BRdata instances, which include larger-scale and more complex problems, the MIP-EGO configurator [2] is adopted to tune mutation probabilities to find the best parameter values for each problem. The hypervolume indicator of the solution set has been used in MIP-EGO as the objective value to tune algorithm configuration. The basic parameter settings of MIP-EGO are a 200 evaluation budget, random forest surrogate model, MIES as internal optimizer and an ordinal search space. With the best parameter setting of the mutation probabilities for BRdata instances, we compare our experimental results with the reference set in [5], which is formed by gathering all non-dominated solutions found by all the implemented algorithms in [5] and also non-dominated solutions from other state-of-the-art MOFJSP algorithms, the proposed algorithm found previously unknown non-dominated solutions.

Another comparison is between our algorithm (FJSP-MOEA) and MOGA [3], SEA [1] and MA1, MA2 [5]. Table 1 displays the hypervolume value of the Pareto front (PF) approximations from all algorithms and the new reference set which is formed by combining all solutions from the PFs by all algorithms. The highest hypervolume value on each problem in all algorithms has been highlighted in bold. We observe that FJSP-MOEA and MA1, MA2 show the best and similar performance, and MOGA behaves the best for three of the BRdata instances. The good performance of MOGA on three problems is interesting. MOGA has an entropy-based mechanism to maintain decision space diversity which might be beneficial for solving these problem instances. For Kacem instances and with fixed mutation probabilities, our obtained non-dominated solutions are the same as the PF in the reference set. MA1 and MA2 also achieved the best PF for all Kacem instances, but our algorithm uses far less computational resources. The proposed FJSP-MOEA uses only a population size of 100 whereas the population size of MA algorithms is 300. FJSP-MOEA uses only 10, 000 objective function evaluations, whereas MA used 150, 000 evaluations.

Table 1: Hypervolume obtained by MOEAs and reference set

Probs	MOGA	SEA	MA1	MA2	FJSP-MOEA	Ref
Mk01	0.00426	0.00508	**0.00512**	**0.00512**	**0.00512**	0.00512
Mk02	0.01261	0.01206	**0.01294**	**0.01294**	**0.01294**	0.01294
Mk03	**0.02460**	0.02165	0.02165	0.02165	0.02165	0.02809
Mk04	**0.06906**	0.06820	0.06901	0.06901	0.06901	0.07274
Mk05	0.00626	0.00635	**0.00655**	**0.00655**	**0.00655**	0.00655
Mk06	0.05841	0.06173	0.06585	0.06692	**0.06709**	0.07065
Mk07	0.02244	0.02132	**0.02269**	**0.02269**	**0.02269**	0.02288
Mk08	**0.00418**	0.00356	0.00361	0.00361	0.00361	0.00428
Mk09	0.01547	0.01755	0.01788	**0.01789**	0.01785	0.01789
Mk10	0.01637	0.01778	0.02145	**0.02196**	0.02081	0.02249

In conclusion, a novel MOEA for MOFJSP is proposed. It uses multiple initialization approaches to enrich the first-generation population, and various crossover operators to create better diversity for offspring. Moreover, to determine the optimal mutation probabilities, the MIP-EGO configurator is adopted to automatically generate proper mutation probabilities. Besides, the straightforward local search is employed with different levels to aid more accurate convergence to the PF. The proposed customization approach can in principle be combined with almost all MOEAs. In this paper, we incorporate it with NSGA-III, to solve MOFJSP. The new algorithm show state-of-the-art performance with less computing budget and can even find new Pareto optimal solutions for the large scale instances.

ACKNOWLEDGMENTS

This work is part of the research programme Smart Industry SI2016 with project name CIMPLO and project number 15465, which is (partly) financed by NWO.

REFERENCES

[1] Tsung-Che Chiang and Hsiao-Jou Lin. 2013. A simple and effective evolutionary algorithm for multiobjective flexible job shop scheduling. *International Journal of Production Economics* 141, 1 (2013), 87–98.

[2] Bas van Stein, Hao Wang, and Thomas Bäck. 2018. Automatic configuration of deep neural networks with EGO. *arXiv preprint arXiv:1810.05526* (2018).

[3] Xiaojuan Wang, Liang Gao, Chaoyong Zhang, and Xinyu Shao. 2010. A multi-objective genetic algorithm based on immune and entropy principle for flexible job-shop scheduling problem. *The International Journal of Advanced Manufacturing Technology* 51, 5-8 (2010), 757–767.

[4] Yali Wang, Bas van Stein, Michael Emmerich, and Thomas Bäck. 2020. A Tailored NSGA-III Instantiation for Flexible Job Shop Scheduling. *arXiv preprint arXiv:2004.06564* (2020).

[5] Yuan Yuan and Hua Xu. 2013. Multiobjective flexible job shop scheduling using memetic algorithms. *IEEE Transactions on Automation Science and Engineering* 12, 1 (2013), 336–353.