CHEMICAL ENGINEERING TRANSACTIONS

VOL. 103, 2023

A publication of AIDIC

The Italian Association of Chemical Engineering
Online at www.cetjournal.it

Guest Editors: Petar S. Varbanov, Panos Seferlis, Yee Van Fan, Athanasios Papadopoulos
Copyright © 2023, AIDIC Servizi S.r.l.
ISBN 979-12-81206-02-1; **ISSN** 2283-9216

DOI: 10.3303/CET23103065

A Hybrid Framework Integrating Machine-learning and Mathematical Programming Approaches for Sustainable Scheduling of Flexible Job-shop Problems

Dan Li[a], Taicheng Zheng[a], Jie Li[a,*], Aydin Teymourifar[b]

[a]Centre for Process Integration, Department of Chemical Engineering, The University of Manchester, M13 9PL, United Kingdom
[b]CEGE - Centro de Estudos em Gestão e Economia, Católica Porto Business School, Porto, Portugal
jie.li-2@manchester.ac.uk

Flexible job shop scheduling has received considerable attention due to its extensive applications in manufacturing. High-quality scheduling solutions are desired but hard to be guaranteed due to the NP-hardness of computational complexity. In this work, a novel energy-efficient hybrid algorithm is proposed to effectively address the scheduling of flexible job shop problems within reasonable time frames. The hybrid framework hybridizes gene expression programming, variable neighborhood search, and simplified mixed integer linear programming approaches to minimize the total energy consumption. It is utilized to address 20 benchmark examples with moderate- or high-complexities. Computational results show that the hybrid algorithm can reach optimality for all considered moderate-size examples within two seconds. The proposed algorithm demonstrates significant competitive advantages relative to the existing mathematical programming approaches and a group-based decomposition method. Specifically, it shortens the computational time over one order of magnitude in some cases and leads to lower total energy consumption with a maximum decrease by 14.5 %.

1. Introduction

As environmental concepts awaken, incremental numbers of process industries pursue sustainable manufacturing (Rakovitis et al., 2020) by developing energy-efficient scheduling approaches to optimize energy-oriented objectives, such as minimizing total energy consumption (TEC), energy costs, or carbon emissions. Flexible job-shop scheduling problems (FJSP) gather significant attention from investigators in various fields, like enterprise management, aerospace, and semiconductor manufacturing (Chaudhry and Khan, 2015). Heuristic dispatching rules (DRs), such as shortest processing time and first in first out, superior on easy implementation and short time requirement (Kaban et al., 2012). They are commonly used by facilities to address the FJSP with economic-oriented or time-oriented objectives. However, these rules are hard to guarantee to yield high-quality solutions to various problems and seldom aim to optimize energy consumption. Solution strategies based on artificial intelligence have been demonstrated (Xie et al., 2019) to generate better solutions than the heuristic rules because the qualities of dispatching rules are improved by the algorithm's self-learning and training. More importantly, artificial intelligence-based approaches so the scheduling problems in a reasonable time frame.
Gene expression programming (GEP) variants have been proven capable of extracting efficient dispatching rules and exploring competitive solutions in a short time frame for FJSP. However, they always fail to reach optima, even in small- or moderate-scale instances. This can be validated by the comparisons between results from the GEP (Zhang et al., 2017) and a mathematical programming approach (Rakovitis et al., 2022). Results manifest that mathematical formulation and GEP are superior in solution accuracy and computational efficiency. A hybrid algorithm integrating these two kinds of approaches is promising to combine their advantages and generate even good solutions in reasonable computational time. Attempts to hybridize artificial intelligent algorithms (e.g., a genetic algorithm) and exact algorithms (e.g., MILP model) are made by Paul et al. (2022) – optimizing batteries, and Ribas et al. (2013) – for pipeline scheduling. Their investigations reveal the superiority

Paper Received: 30 April 2023; Revised: 27 June 2023; Accepted: 22 July 2023

of the hybrid algorithm relative to single algorithms. In this work, we aim to develop a hybrid approach to integrate multiple algorithms for scheduling FJSP with energy-oriented objectives.

2. Problem statement

A FJSP example involves a set of jobs $k \in \mathbf{K} = \{1,2,3,\dots,K\}$ and machines $j \in \mathbf{J} = \{1,2,3,\dots,J\}$. Each job k is required to be processed through a sequence of operations $i \in \mathbf{I}_k$, whose number is predefined but not necessarily identical among jobs. Machines being able to operate i are included in a set $\mathbf{J}_i$. The optimization objective is to minimize TEC, including the direct, indirect and unload energy consumption. Parameters of FJSP are given as follows. An operation i of job k processed on a machine j are featured by its processing time (P_{kij}^{T}) and cutting power (P_{kij}^{C}). To save energy, a machine j during idle slots selects a machine mode between switch off-on mode with energy consumption E_j^{O} and standby mode consuming unload power (P_j^{U}) per unit time. It is assumed that all parameters are deterministic. Jobs are released at the beginning of the scheduling horizon. In addition, machines are switched on/off exactly before/after the first/last operations processed on them.

3. Hybrid algorithm

The proposed hybrid algorithm (HA) is elaborately described in this section. Three components, including GEP, variable neighborhood search (VNS) and Mixed Integer Linear Programming (MILP) formulation, are integrated as illustrated in Figure 1. Scheduling solutions generated from the optimizing algorithm in each step are transferred to the successor step as input, through which the searching process advances in a positive direction and solution qualities improve incrementally. A parallel computing approach is applied to effectively promote computational efficiency. Specifically, the master-slave parallel model and asynchronous parallel computing mode (Sevkli and Aydin, 2007) for job shop scheduling and (Akbay et al., 2020) for portfolio optimization are implemented to parallelize the GEP and VNS. Parallel computing is implemented for all MILP formulations by setting 'option threads=0' in GAMS (2023).

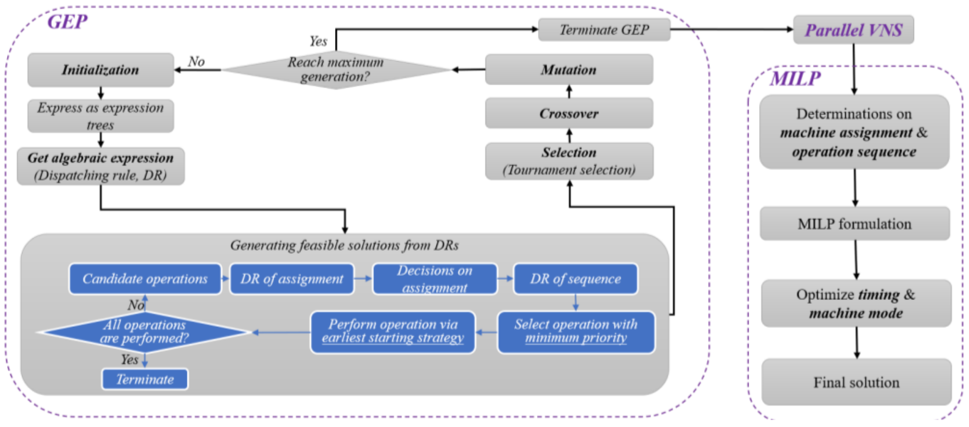
GEP
Terminate GEP
Parallel VNS
Yes
Initialization
No
Reach maximum generation?
Mutation
Express as expression trees
Crossover
Get algebraic expression (Dispatching rule, DR)
Selection (Tournament selection)
Generating feasible solutions from DRs
Candidate operations
DR of assignment
Decisions on assignment
DR of sequence
No
All operations are performed?
Perform operation via earliest starting strategy
Select operation with minimum priority
Yes
Terminate
MILP
Determinations on machine assignment & operation sequence
MILP formulation
Optimize timing & machine mode
Final solution

Figure 1: Framework of the hybrid algorithm

3.1 Gene expression programming

The DRs extracted for assignment and sequence are supposed to be discrepant as different decisions have various sensitivities on identical attributes. In the designed GEP, DRs for machine assignment and operation sequences are represented using different parts in one chromosome, illustrated using DR-A and DR-S. They have identical features and structures, being constructed using the same approach, but may vary in the gene of each position due to randomness. The multigene structure is utilized to construct chromosomes, as exemplified by Figure 2, which presents the part of the chromosome for machine assignment. Each gene partitioned into a head and a tail, is composed of functions and terminals with fixed lengths of symbols. Function sets in all genes comprise of five operators ($+, -, \times, \div, \sqrt{}$). Similar to the work of Zhang et al., (2017), there are five attributes (i.e. processing time P_{kij}^{T}, cutting power P_{kij}^{C}, unload power P_j^{U}, idle time Dr_j, number of unscheduled operations N_{ki}^{r}), which are closely associated with TEC, included in the terminal sets. Different genes are connected using linking functions, which are represented using a linking gene (see the last gene in Figure 2). The terminal set in the linking gene is different from those in other genes, where terminals denote the ordinal numbers of genes in the chromosome. Corresponding dispatching rules could be constructed from the chromosome using depth-first decoding approach (Yang et al., 2016).

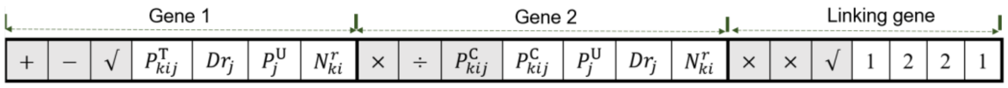
Gene 1
Gene 2
Linking gene
$+$ $-$ $\surd$ P^{T}_{kij} Dr_j P^{U}_j N^r_{ki} $\times$ $\div$ P^{C}_{kij} P^{C}_{kij} P^{U}_j Dr_j N^r_{ki} $\times$ $\times$ $\surd$ 1 2 2 1

Figure 2: An example of chromosome for assignment (DR-A) with two genes and a linking gene

The evolutionary process of GEP is illustrated in Figure 1. Genetic operators include selection, crossover, and mutation to screen individuals with good adaptability, inherit metrics and maintain diversity. The tournament selection strategy is utilized here to make comparisons between two candidate individuals and reserve the better one with lower TEC. The comparison and selection repeat until the new population composed of selected individuals have an identical size to the parent population. Adaptive crossover and mutation (Chen et al., 2019) are adopted to improve genetic efficiency and prevent algorithm premature even stagnation. Adaptive crossover rate C^{r}_g and mutation rate M^{r}_g at current generation g are calculated using Eqs.(1)-(2), with $C^{\mathrm{r}}_0 = 0.6$ and $M^{\mathrm{r}}_0 = 0.3$. Variables TEC^1_g, TEC^2_g and TEC^3_g express the average, minimum and maximum values of TEC, among all individuals in current generation g. Evolutionary process of GEP stops until the maximum generation is reached, as pictured in Figure 1.

$$C^{\mathrm{r}}_g = C^{\mathrm{r}}_0 \cdot \left(1 + 0.2 \cdot \frac{g}{g^{\max}} \cdot \frac{{TEC^1_g}^2}{\left(TEC^3_g - TEC^2_g\right)^2 + {TEC^1_g}^2}\right) \quad \forall g \tag{1}$$

$$M^{\mathrm{r}}_g = M^{\mathrm{r}}_0 \cdot 1.2^{\left(\frac{(g-1)\cdot TEC^1_g}{(TEC^3_g - TEC^2_g) + TEC^1_g}\right)^{0.3}} \quad \forall g > 1 \tag{2}$$

3.2 Variable neighborhood search

VNS algorithm, as an extraordinary local-search approach, exploits promising solutions near the current dominated solutions. It hybridizes with GEP in serial and improves solution qualities starting from elite solutions given by GEP. A knowledge-guided NS is designed to adjust the machine assignment of the current solution. In detail, experiences on machine assignment are learned from elite solutions in earlier iterations, and the knowledge feeds back to the search process to locate promising areas, which is inspired by the work of Zheng et al. (2016). Probabilities Pro^g_{ij} of assigning unit j to an operation i at the iteration g is stored in the knowledge base. The probabilities are initialized as Eq(3) at $g = 0$. And they update in each iteration using Eq(4), where $x^{\mathrm{L}}_{egij} = 1$ if i is assigned to j for the elite solution e at iteration g. Eq(5) makes normalization for all machines of one operation. To find new nneighboringsolutions, the machine j for i would be mutated to another j': $(j' \in \mathbf{J}_i, j' \neq j)$ that is selected from the knowledge base using roulette wheel method.

$$Pro^{g=0}_{ij} = \begin{Bmatrix} \frac{1}{|\mathbf{J}_i|} & j \in \mathbf{J}_i \\ 0 & j \notin \mathbf{J}_i \end{Bmatrix} \tag{3}$$

$$Pro^g_{ij} = Pro^{g-1}_{ij} + 0.5 \sum_e x^{\mathrm{L}}_{egij} \tag{4}$$

$$Pro^g_{ij} = \frac{Pro^g_{ij}}{\sum_{j' \in \mathbf{J}_i} Pro^g_{ij'}} \tag{5}$$

3.3 Mixed Integer Linear Programming approach

Solutions generated from the artificial intelligent algorithms would be optimized using a local sequence-based MILP formulation to adjust the decisions on timing and machine mode (see Figure 1). Assigning decisions of machine j to an operation i of job k and sequencing relations between two operations on the same machine are expressed using parameters W_{kij} and $X_{kik'i'j}$. That is $W_{kij} = 1$ if an operation i of job k is processed by a machine j, and $X_{kik'i'j} = 1$ if operation i of job k immediately precedes operation i' of job k' on machine j. We define sets of binary (z_{kij}) and continuous (y_{kij}) variables to express the machine mode of switch off-on and standby. If a machine j is switched off after processing an operation i of job k, the variable z_{kij} equals to one (i.e. $z_{kij} = 1$). Otherwise, $z_{kij} = 0$ and $y_{kij} = 1$, as handled by Eq(6). Set $\mathbf{I}_{kj}$ includes operations $i \in \mathbf{I}_k$ being able to be processed on a machine j.

$$y_{kij} + z_{kij} = W_{kij} \quad \forall k, j, i \in \mathbf{I}_{kj} \tag{6}$$

One operation i that is the successor operation of i' in the same job (i.e. $i' \in \mathbf{I}_k, i \in \mathbf{I}_k, i = i' + 1$), is enforced to start after the finish of i'. Variables T_{ki} expresses the start time of i in job k.

$$T_{ki} \geq T_{ki'} + \sum_{j \in \mathbf{J}_{ki'}} \left(P^{\mathrm{T}}_{ki'j} \cdot W_{ki'j}\right) \quad \forall k, i' \in \mathbf{I}_k, i \in \mathbf{I}_k, i = i' + 1 \tag{7}$$

A continuous variable ST_{kij} denotes the duration of standby mode for a machine j after processing $i \in \mathbf{I}_k$. It equals the time interval between the operation i and its immediate precedence operation $i' \in \mathbf{I}_{k'}$ on j.

$$T_{k'i'} \geq T_{ki} + \sum_{j \in \mathbf{J}_{ki}} \left(P^{\mathrm{T}}_{kij} \cdot W_{kij} + ST_{kij}\right) - H \cdot \left(1 - \sum_{j \in (\mathbf{J}_{ki} \cap \mathbf{J}_{k'i'})} X_{kik'i'j}\right) \quad \forall k, k', i \in \mathbf{I}_k, i' \in \mathbf{I}_{k'}, \exists j \in (\mathbf{J}_{ki} \cap \mathbf{J}_{k'i'}) \tag{8}$$

$$T_{k'i'} \leq T_{ki} + \sum_{j \in \mathbf{J}_{ki}} \left(P^{\mathrm{T}}_{kij} \cdot W_{kij} + ST_{kij}\right) + H \cdot \left(1 - \sum_{j \in (\mathbf{J}_{ki} \cap \mathbf{J}_{k'i'})} X_{kik'i'j}\right) + H \cdot \sum_{j \in \mathbf{J}_{ki}} z_{kij} \quad \forall k, k', i \in \mathbf{I}_k, i' \in \mathbf{I}_{k'}, \exists j \in (\mathbf{J}_{ki} \cap \mathbf{J}_{k'i'}) \tag{9}$$

Standby energy consumption ES_{ki} is proportional to the duration of standby time and the unload power of the machine j. Variable ST_{kij} is enforced as zero if the machine does not remain standby after processing $i \in \mathbf{I}_k$.

$$ES_{ki} = \sum_{j \in \mathbf{J}_{ki}} \left(ST_{kij} \cdot P^{\mathrm{U}}_j\right) \quad \forall k, i \in \mathbf{I}_k \tag{10}$$

$$ST_{kij} \leq min(H, \frac{E^{\mathrm{O}}_j}{P^{\mathrm{U}}_j}) \cdot y_{kij} \quad \forall k, j, i \in \mathbf{I}_{kj} \tag{11}$$

Makespan is larger than the finish of the last operation ($\mathbf{LI}_k$) in all jobs, as Eq(12). In any machine, accumulated time used to process operations and keep standby is supposed to be no larger than the makespan.

$$T_{ki} + \sum_{j \in \mathbf{J}_{ki}} \left(W_{kij} \cdot P^{\mathrm{T}}_{kij} + ST_{kij}\right) \leq MS \quad \forall k, i \in (\mathbf{I}_k \cap \mathbf{LI}_k) \tag{12}$$

$$\sum_{k} \sum_{i \in \mathbf{I}_{kj}} \left(W_{kij} \cdot P^{\mathrm{T}}_{kij} + ST_{kij}\right) \leq MS \quad \forall j \tag{13}$$

We aim to minimize TEC, where direct energy consumption ($P^{\mathrm{T}}_{kij} \cdot P^{\mathrm{C}}_{kij}$) is attributed to processing operations, indirect energy consumption ($\beta \cdot MS$) is required by auxiliary facilities or subsidiary sectors (e.g. lighting and air conditioning), and the unload energy is consumed by machines during idle slots.

$$TEC = \sum_{j} \sum_{k} \sum_{i \in \mathbf{I}_{kj}} \left(W_{kij} \cdot P^{\mathrm{T}}_{kij} \cdot P^{C}_{kij}\right) + \beta \cdot MS + \sum_{k} \sum_{i \in \mathbf{I}_k} ES_{ki} + \sum_{j} \sum_{k} \sum_{i \in \mathbf{I}_{kj}} \left(E^{\mathrm{O}}_j \cdot z_{kij}\right) \tag{14}$$

An operation should start later than the minimum required time for all predecessor operations in the same job.

$$T_{ki} \geq \sum_{i' < i, i' \in \mathbf{I}_k} \min_{j \in \mathbf{J}_{ki'}} \left(P^{\mathrm{T}}_{ki'j}\right) \quad \forall k, i \in \mathbf{I}_k \tag{15}$$

Eqs(16)-(17) list all the continuous and binary variables of the model. We complete the development of the mathematical model M, which serves as the third-step optimization in the hybrid framework. The local sequence-based mathematical model M comprises of Eqs(6)-(17).

$$ES_{ki} \geq 0, ST_{kij} \geq 0, T_{ki} \geq 0, 1 \geq y_{kij} \geq 0 \tag{16}$$

$$z_{kij} \in \{0, 1\} \tag{17}$$

4. Computational results

We now evaluate the computational performance of the designed HA and assess its capability on exploring high-quality solutions. There are 20 benchmark examples (i.e. Examples 1-20 in Table 1) originated from works of Zhang et al. (2017) – emphasizing evolutionary generation and Rakovitis et al. (2022) further increasing energy efficiency. Specifically, Examples 1-10 in this work correspond to the Examples 1-10 from Rakovitis et al., (2022). Examples 10-20 with high complexities are the Examples 41-50 from Rakovitis et al., (2022). Comparative studies are conducted relative to existing algorithms referred from literature and the model M proposed in this work. The unit-specific event-based MILP formulation and grouping-based decomposition approach proposed by Rakovitis et al., (2022) are abbreviated as MU and GD. All algorithms are implemented and run using a desktop computer with AMD RyzenTM 9 3900X 3.8 GHz and 48 GB RAM running Windows 10. Parallel computing is implemented for HA and all MILP approaches (i.e. M and MU) with 12 cores. In the spirit of fair comparisons, all algorithms address identical examples without additional pre-processing.

Table 1 reveals the computational results of the proposed HA and the literature best results among MU and GD. Here, the relative gaps disclose the relative deviations between the TEC results from HA and literature approaches, and a negative value of the relative gap evinces HA yields better scheduling solutions with lower TEC. We observe the same objective results from HA versus the literature algorithms for moderate-size examples (i.e., Examples 1-10) listed in Table 1. The proposed HA is capable of finding the global optima, which are validated by the exact approach MU. With higher complexities, HA has been demonstrated to have significant strengths in decreasing energy consumption with a maximum reduction of 14.5 % (6, 670.1 kW vs. 7,800.2 kW in Example 20). More importantly, computational times are reduced by an approximate factor of 10 (e.g. 305 s vs. 3,600 s) while addressing examples 12,13 and 15. Additionally, relative to MU, we do not need to predefine the event points, and as a result an iterated procedure is not required to find an appropriate number of event points to generate the optimality, which exerts excessive computational burdens for addressing industrial-scale problems.

Table 1. Computational results of benchmark examples from HA and literature existing algorithms (Zhang et al., 2017; Rakovitis et al., 2022)

Example	HA		Existing algorithms		
	TEC (kW)	CPU time (s)	TEC (kW)	CPU time (s)	Relative Gap (%)
1	63.03	1.2	63.03	0.1	0
2	122.44	1.1	122.44	0.1	0
3	75.74	1.1	75.74	0.1	0
4	146.63	1.1	146.63	0.1	0
5	78.40	1.1	78.40	0.1	0
6	220.74	1.1	220.74	0.1	0
7	97.54	1.1	97.54	0.2	0
8	146.81	1.0	146.81	0.1	0
9	230.66	1.1	230.66	0.2	0
10	161.06	1.1	161.06	0.2	0
11	3,475.7	303	3,565	301	-2.5
12	3,582.8	305	3,635.8	3,600	-1.5
13	3,726.8	309	3,884.4	3,600	-4.1
14	5,122.8	368	5,357.5	412	-4.4
15	4,654.9	356	4,899.7	3,600	-5.0
16	5,016.5	353	5,256.1	304	-4.6
17	4,941.5	354	5,033.9	401	-1.8
18	5,038.3	357	5,595.8	400	-10.0
19	6,238.3	413	6,946.2	1.2	-10.2
20	6,670.1	410	7,800.2	0.8	-14.5

Table 2 reports the comparative studies between the hybrid algorithm and the single GEP or exact algorithm M, where NA means no available solutions are found within the time resource limitation. We can observe that the hybrid algorithm always shows better performance. In terms of the solution quality, HA results in better balance between exploitation and exploration features, manifesting better objective results (i.e. lower TEC) certainly generated relative to GEP. Concerning the computational efficiency, HA leads us to decrease the computational time over one order of magnitude relative to model M with saving the energy consumption of 577 kW (5,016.5 kW vs. 5,593.6 kW). One can conclude that combining multiple approaches, it is evident to benefit from them to generate significantly better solutions in less computational time.

Table 2 Comparative studies among algorithms HA and GEP, M

Example	HA		GEP		M	
	TEC (kW)	CPU time (s)	TEC (kW)	CPU time (s)	TEC (kW)	CPU time (s)
11	3,475.7	303	3,674.1	32	3,433.1	3,600
12	3,582.8	305	3,710.5	31	3,472.2	3,600
13	3,726.8	309	4,089.0	32	3,525.9	3,600
14	5,122.8	368	5,337.8	49	5,274.2	3,600
15	4,654.9	356	4,927.7	49	4,818.9	3,600
16	5,016.5	353	5,392.6	48	5,593.6	3,600
17	4,941.5	354	5,144.0	47	5,305.4	3,600
18	5,038.3	357	5,517.7	50	5,144.3	3,600
19	6,238.3	413	6,501.2	68	NA	3,600
20	6,670.1	410	6,954.6	72	NA	3,600

5. Conclusions

In this work, we present a hybrid algorithm hybridizing artificial intelligence and the mathematical programming approach to expose the energy-oriented dispatching rules and yield good-quality solutions. Exact algorithms are integrated to approach optima by adjusting decisions on timing and machine modes. Case studies demonstrated that the hybrid algorithm can reach optimality for all considered moderate-complexity examples within seconds and lower energy consumptions (with a maximum of 14.5 %) using significantly reduced computational time (by a factor of 10) in complicated instances relative to the existing exact and decomposition approaches.

Acknowledgments

Dan Li and Taicheng Zheng appreciate the financial support from China Scholarship Council - the University of Manchester Joint Scholarship (201908130170, 202106440020). Jie Li appreciates financial support from Engineering and Physical Sciences Research Council (EP/T03145X/1, EP/V051008/1).

References

Akbay M.A., Kalayci C.B., Polat O., 2020, A parallel variable neighborhood search algorithm with quadratic programming for cardinality constrained portfolio optimization. Knowledge- Based Systems, 198, 105944.

Chaudhry I.A., Khan A.A., 2016, A research survey: review of flexible job shop scheduling techniques. International Transactions in Operational Research, 23(3), 551-591.

Chen X., Zhang B., Gao D., 2019, Algorithm based on improved genetic algorithm for job shop scheduling problem. 2019 IEEE International Conference on Mechatronics and Automation (ICMA), 04-07 August 2019, 951-956, DOI: 10.1109/ICMA.2019.8816334.

GAMS, 2023, System Overview. <https://www.gams.com/products/gams/gams-language/>, accessed 24/07/2023.

Kaban A.K., Othman Z., Rohmah D.S., 2012, Comparison of dispatching rules in job-shop scheduling problem using simulation: a case study. International Journal of Simulation Modelling, 11(3), 129-140.

Paul S., Ganguly B., Adhikary U., 2022, A Hybrid MILP-GA Algorithm to Optimize Battery Mix System in Active Distribution Networks. 2022 IEEE VLSI Device Circuit and System (VLSI DCS). IEEE, 123-128.

Rakovitis N., Li D., Zhang N., Li J., Zhang L., Xiao X., 2020, Novel Approaches for Energy-Efficient Flexible Job-Shop Scheduling Problems. Chemical Engineering Transactions, 81, 823-828.

Rakovitis N., Li D., Zhang N., Li J., Zhang L., Xiao X., 2022, Novel approach to energy-efficient flexible job-shop scheduling problems. Energy, 238, 121773.

Ribas P.C., Yamamoto L., Polli H.L., Arruda L.V., Neves-Jr F., 2013, A micro-genetic algorithm for multi-objective scheduling of a real world pipeline network. Engineering Applications of Artificial Intelligence, 26, 302-313.

Sevkli M., Aydin M.E., 2007, Parallel variable neighbourhood search algorithms for job shop scheduling problems. IMA Journal of Management Mathematics, 18(2), 117-133.

Xie J., Gao L., Peng K., Li X., Li H., 2019, Review on flexible job shop scheduling. IET Collaborative Intelligent Manufacturing, 1, 67-77.

Yang Y., Li X., Gao L., Shao X., 2016, Modeling and impact factors analyzing of energy consumption in CNC face milling using GRASP gene expression programming. The International Journal of Advanced Manufacturing Technology, 87, 1247-1263.

Zhang L., Tang Q., Wu Z., Wang F., 2017, Mathematical modeling and evolutionary generation of rule sets for energy-efficient flexible job shops. Energy, 138, 210-227.

Zheng X.L., Wang L., 2016, A knowledge-guided fruit fly optimization algorithm for dual resource constrained flexible job-shop scheduling problem. International Journal of Production Research, 54(18), 5554-5566.