

Title	Energy-Efficient Multi-Objective Hybrid Flowshop Scheduling Problem with Blocking Constraints
Authors	Missaoui, Ahmed;Ozturk, Cemalettin;O'Sullivan, Barry
Publication date	2023-07
Original Citation	Missaoui, A., Ozturk, C. and O'Sullivan, B. (2023) 'Energy-efficient multi-objective hybrid flowshop scheduling problem with blocking constraints', in 2023 9th International Conference on Control, Decision and Information Technologies (CoDIT), Rome, Italy, 03-06 June. IEEE, (6 pp). https://doi.org/10.1109/CoDIT58514.2023.10284309
Type of publication	Article (peer-reviewed);Conference item
Link to publisher's version	https://doi.org/10.1109/CoDIT58514.2023.10284309 - 10.1109/codit58514.2023.10284309
Rights	© 2023 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works
Download date	2026-10-26 21:24:57
Item downloaded from	https://hdl.handle.net/10468/15183



UCC

University College Cork, Ireland
 Coláiste na hOllscoile Corcaigh

Energy-efficient multi-objective hybrid flowshop scheduling problem with blocking constraints

1st Ahmed Missaoui
Insight Centre for Data Analytics
University College Cork
Cork, Ireland
ahmed.missaoui@insight-centre.org

2nd Cemalettin Ozturk
Process, Energy Transport Engineering
Munster Technological University
Cork, Ireland
cemalettin.ozturk@mtu.ie

3rd Barry O'Sullivan
School of Computer Science & IT
University College Cork
Cork, Ireland
b.osullivan@cs.ucc.ie

Abstract—The social context in relation to energy policies, energy supply, and sustainability concerns, as well as advances in more energy-efficient technologies is driving a need for a change in the manufacturing industry. In this paper, a hybrid flowshop production system with blocking constraints is investigated from both sustainability and productivity dimensions. For an efficient solution to this proven NP-hard problem, we implemented a multi-objective iterated greedy algorithm first time to minimize the makespan and total energy consumption simultaneously. We provide small, medium, and large instances to benchmark our approach against non-dominated sorting genetic algorithm-II (NSGA-II) which shows efficiency of the proposed methodology.

Index Terms—Multi-objective iterated greedy; hybrid flowshop scheduling with blocking constraint; energy efficiency; makespan, NSGA-II, Pareto front.

I. INTRODUCTION

The manufacturing industry accounted for 25.6% of the energy consumed within the EU [1]. Given the dependency on fossil fuel resources, ongoing global instability threatening energy security, as well as sustainable development goals [2] necessitate innovative methods to reduce energy consumption. While there are substantial amount of work to make manufacturing equipment and machining more energy efficient, research shows that the majority of the energy waste is observed during non-value added operations such as machine idle time, changeover and setup time and blocking time. All these issues can be taken into account in addition to the productivity objectives.

Hence, considering both energy and productivity concerns in system level and developing innovative manufacturing scheduling methods focusing both has great potential to reduce energy consumption and increase productivity simultaneously.

Pinedo's book ([3]) provides a general classification of machine scheduling problems based on arrangement of machines and the workflow. According to this classification, one of the common production system is flowshop scheduling problem (FSP) that was introduced by Johnson ([4]). In FSP, there is a set of jobs that have to be processed in more than one production stage and each of them comprising only one machine.

A very practical extension of FSP, the hybrid flowshop scheduling problem (HFS) was introduced to the literature by

Gupta ([5]) which considers parallel machines in every stage of FSP. In this work, author proved that even for the most simple instance with two production stages, the first with one machine and the second only with two parallel machines, the problem is NP-hard [5]. However, Meanwhile, HFS is very significant in many different manufacturing industries such as metal [6], glass [7], automobile manufacturing [8] and pharma [9].

In some industries, the category of product and some environmental factors impose some constraints that have to be respected and taken into account. For instance, the limitations of storage capacity between stages force finished jobs in a given stage to wait if the next stage is not available. In other words a machine in upstream stage cannot release the job after its processing finished until there is an available machine in downstream stage. A detailed survey of machine scheduling problems with blocking constraints is provided in [10]. The blocking HFS (BHFS) is widely studied with different criteria such as total earliness tardiness: in [11], Makespan: in [12], Total energy consumption (TEC): in [13]. As seen, most works are with a single objective either productivity or energy consumption and simultaneous consideration of them for HFS production systems is not studied in the literature. Due to its significant practicality, there is a great potential in investigating multi-objective approaches to attack this problem.

There are several multi-objective scheduling methods developed for scheduling problem such as the weighted sum method in which a weight is assigned for each objective according to its importance [14]. If the relative importance of each objective is known apriori, the lexicographic method can be used [15], but if it is not possible to distinguish the relative importance of objectives, Pareto optimization [16] method can be implemented which is a trade-off between objectives. To the best of our knowledge, this is the first work investigating the bi-objective BHFS problem with identical parallel machines aiming to reduce the TEC and the makespan to find Pareto front for both objectives. Result of our study is providing effective decision support to the decision makers who can select from several equal quality solutions based on their preferences.

The remainder of the paper is organized as follows: in Section II we provide a survey of the relevant literature. In

Section III, the problem is described in detail. The proposed resolution method is explained in Section IV. Benchmarking method used in evaluating the performance of the proposed algorithm is described in Section V. In Section VI, we provide computational experiments. Finally, we conclude our study and highlight future work directions in Section VII.

II. LITERATURE REVIEW

Energy-efficient scheduling refers to the set of tools and algorithms used to optimize energy consumption in production systems taking into account several energy-related aspects by using various solution methods. Energy efficient HFS literature finds its roots back in [17] where authors investigated a case study of HFS in steel industries aiming to minimize the total completion time under power constraints. It is observed that the number of publications related to energy-efficient scheduling has been increasing in the last few years. Authors in [18] presented an early example of a mixed integer programming model (MIP) and a genetic algorithm (GA) for minimizing total energy cost on a shop floor considering the start-up and shut-down aspects.

In another work, [19], authors proposed a MIP model and a heuristic for the bi-objective flexible flowshop aiming to minimize makespan and total tardiness taking into consideration energy saving. [20] developed a multi-objective ant colony optimization methods for a HFS with makespan and electric power cost under time-of-use tariff (TOU). [21] presents the same problem with total energy cost using a simulated annealing algorithm. The multi-objective HFS with release dates was tackled in [22]. In this paper, an evolutionary algorithm based on NSGA-II is used to get the Pareto front of the energy saving presented in the total waiting time between operations, earliness, tardiness, and product quality. [23] describes an energy-aware optimization algorithm for multi-objective HFS with setup times in order to minimize the makespan and energy consumption.

It can be seen that energy-aware HFS has been investigated with different aspects in the last few years. Moreover, several constraints have been studied in energy-efficient scheduling domain. "Blocking constraints" is one of the which has many application areas in industry. Authors in [24] describes a MIP model with peak power for an FSP with blocking constraint, in order to minimize makespan. The same problem is tackled in [25] when makespan and TEC criteria are considered using a set of evolutionary metaheuristics based on GA. Very recently in [26], authors introduced a Pareto multi-objective optimization model for the distributed FSP with blocking constraints.

Although there are several studies in the literature considering blocking constraints in energy-efficient scheduling domain, there are only a few studies considering HFS systems. [27] addressed the BHFS. The authors formulate the problem using a MILP model and solve it using a multi-objective objective evolutionary algorithm based on decomposition for the minimization of total tardiness and TEC.

Recently the same problem with a single objective of TEC is tackled in [13], and an effective iterated algorithm (IG) is developed for the resolution. It is clear that population-based metaheuristics are widely applied to address multi-objective energy-efficient scheduling problems. Regardless of this, iterated methods have been proven effective with multi-objective scheduling problems such as in [28] and [29]. As observed, a small number of papers tackled the energy-aware multi-objectives HFS with blocking constraints. Furthermore, to the best of our knowledge, there is no study in the literature that considers both productivity (i.e. makespan) and energy consumption objectives simultaneously for BHFS systems. Hence, in this paper, we will propose our method to resolve this very practical problem. In the following section, we will describe the studied problem in detail.

III. PROBLEM STATEMENT

A. Problem description

As stated above, the HFS problem is widely used to represent manufacturing systems. In fact, it's the most known extension of FSP where there is one stage at least includes two or more resources. Let n be the number of jobs, $j \in 1..n$, that have to be processed in g production stages, where each of them contains m identical machines where $m_i \geq 1$ for all $i \in 1..g$. Using the first available machine (FAM) assignment rule, each job has to be handled by only one machine in every stage and each machine cannot process more than one job at the same time. Buffer capacity between upstream and downstream stages could change the job completion times. In BHFS setting, it is assumed that there is no buffer capacity between stages. Hence, after its completion, each job wait on the machine until one in the next stage is available. The machine in the upstream stage is blocked during this time. The objective is to determine the best order of jobs that is able to optimize one or more criteria specified by the decision-makers. A mixed integer linear programming model minimizing only the energy consumption is introduced in [30]. In this paper we use the same problem definition in [30] with an extended objective function where we minimize both, *Makespan* and Total Energy Consumption, *TEC*, simultaneously. Therefore, below we formulate only the new objective function and refer readers to [30] for the rest of the constraint formulation.

The first objective is the *Makespan*, maximum of job completion times, C_j , while the second objective is total energy consumption during the machine idle time (E_{idle}), blocking time (E_{block}), and processing time (E_{proces}). Both objectives are given in equations (1) and (2) successively.

$$\min Makespan : Cmax = \max\{C_j, j \in n\} \quad (1)$$

$$\min TEC = E_{idle} + E_{Block} + E_{Proces} \quad (2)$$

where

$$E_{idle} = E_i * Idle_{time}$$

$$E_{Block} = E_b * Blocking_{time}$$

TABLE I
PROCESSING TIMES FOR THE EXAMPLE IN HOURS

Job	Stage 1	Stage 2
1	3	4
2	5	3
3	6	2
4	2	2
5	3	2

$$E_{Proces} = E_p * Processing_{time}$$

B. Illustrative example

In this section, we provide an illustrative example for BHFS problem with two stages that each stage has two machines. The processing times of five jobs are given in Table I in hours. The energy consumption E_i , E_b and E_p are 2, 5, and 3 kWh respectively.

The Gantt chart of the above example generated by FAM heuristic is provided in Figure 1. As a result, $C_{max} = 13$ hours and $TEC = 105$ Kwh.

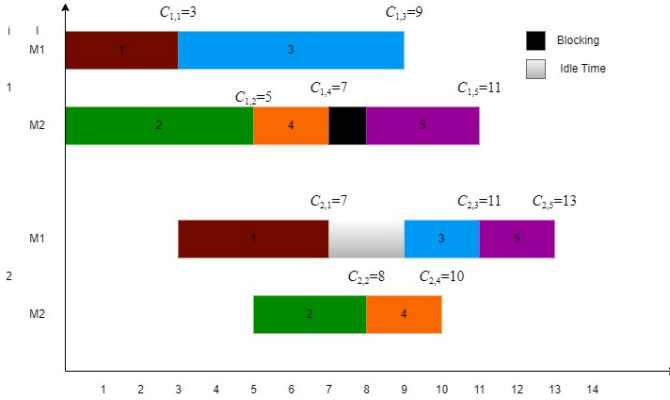


Fig. 1. Gantt chart of Seq (1, 2, 3, 4, 5)

IV. MULTI-OBJECTIVE ITERATED GREEDY ALGORITHM

The iterated greedy algorithm (IG) was first introduced by [31] to solve FSP with makespan criteria. After that, IG has been widely acknowledged as a highly effective method for scheduling problems. IG is extended to multi-objective iterated greedy algorithm (MOIG) in [28] and its efficiency over simulated annealing demonstrated with instances of FSP where the objective is to find the set of Pareto optimal solutions for minimizing makespan and flowtime. In [32], an improved MOIG was introduced to tackle the FSP considering makespan, total tardiness, and flowtime objectives.

MOIG is based on two phases. The first phase is generating the initial Pareto front (FP) by including at least two feasible solutions. In the second phase, the greedy algorithm is applied to each of those solutions to improve the Pareto front by a set of new solutions CS . This process will be repeated until the stopping criterion is met. The outline of MOIG is given in Algorithm 1 where we describe details in the following subsections.

Algorithm 1 Multi-objective Iterated Greedy

```

 $\pi := NEH_{C_{max}}()$ 
 $\pi' := NEH_{Eng}()$ 
 $FP := \{\pi, \pi'\}$ 
while The termination criterion is not reached do
  for  $\pi_i$  in  $FP$  do
     $S_i := GreedyPhase(\pi_i)$ 
     $CS := CS \cup S_i$ 
  end for
   $FP := FP \cup CS$ 
end while
Return  $FP$ 

```

A. Initialization

As in single objective scheduling problems, a good quality set of starting solutions significantly reinforces the quality of IG. As there are two objectives, a solution for each must be initialized. For makespan, we used the original $NEH_{C_{max}}$ heuristic of [33] to find an initial solution, π . This approach is widely used to provide a robust initial solution for makespan criteria. We also use the modified version of this algorithm NEH_{Eng} for the TEC to find an initial solution for this objective criteria, π' using the same principle: Jobs are sorted in the non-ascending order of their total processing time. Then for each partial sequence, all remaining jobs are reinserted in all possible positions and the one with minimum TEC is selected, the same operation is repeated with all jobs. At the end of this initialization phase, an initial Pareto front $FP := \{\pi, \pi'\}$, is found. After this initialisation, the greedy phase is executed as explained in the following section.

B. Greedy phase

The objective of this operator is to generate a new set of solutions via the destruction and reconstruction of the initial one. The first step is to remove a group of d jobs from the sequence π and insert them in a second sequence D knowing that d is a parameter that has to be calibrated.

The second step involves construction, which differs from the single objective IG in that it could give multiple solutions instead of just one at the end. In that case, starting with the insertion of the first job of D in all positions of π , this will generate $n - d + 1$ new partial solutions. We insert all obtained partial sequences in a set of New Partial Sequences, NPS . The next job is then added to all partial sequences using the same process to generate $(n - d + 1) * (n - d + 2)$ new partial sequences.

To avoid an exponential number of solutions in evaluation, after each insertion of one job, only non-dominated solutions will be selected from the NPS set and putted in NDS as candidates for the next job insertion. The process will be repeated until the stopping criterion is satisfied. The outline of the greedy phase is given in algorithm 2.

Algorithm 2 Greedy Phase(π)

```

Remove d jobs from  $\pi$  and put them in set  $D$ 
NDS = ( $\pi$ )
for  $i$  in  $D$  do
  Set  $NWS = Null$ 
  for  $\pi$  Partial sequence in NDS do
    for  $j$  position in ( $\pi$ ) do
       $Seq_{\pi,i,j} :=$  Insertion of  $i$  in  $j$ 
      Insert  $Seq_{\pi,i,j}$  in NPS
    end for
   $NDS :=$  Non dominated solution from  $NWS$ 
end for
end for
Return NDS

```

V. NON-DOMINATED SORTING GENETIC ALGORITHM (NSGA-II)

In this paper, we used well known NSGA-II as a benchmarking method for MOIG algorithm. This metaheuristic is introduced first by [34] to solve a set of multi-objective optimization problems and since then continuously improved and widely used in many combinatorial problem. Particularly, NSGA-II has been used in recent publications on multi-objective energy-efficient scheduling as a high performing competitor for benchmarking, such as in [35]— [36].

NSGA-II principle is basically generating a diverse set of non-dominated solutions through a combination of genetic operators, including population initialization, crossover, and mutation. We adapted this approach to find the Pareto front of BHFS with makespan and TEC objectives as presented in Algorithm 3. Starting with the initial population P_0 , this operator has to create a set of N random permutations and calculate their objective functions. Then, until the termination criteria is met, crossover and mutation operators are used to generate a new population called offspring Q_t . The two populations P_0 and Q_t are merged in R_t and all solutions are evaluated before being sorted on many fronts according to the no-dominance NDS rule. Inside each front, the solutions in turn are sorted according to the crowding distance CD . In accordance with the two ranks, the new initial P_t permutations are selected to create the initial population via the *survivor* operator, and the process is repeated until the stopping criterion is satisfied. The main steps are given in Algorithm 3. NSGA-II has several benefits over other multi-objective optimization algorithms, including a fast aggregation rate and a good variety of solutions.

VI. COMPUTATIONAL EVALUATION AND STATISTICAL EXPERIMENTATION

A. Metrics and Benchmark

Each used algorithm provides a set of non-dominated solutions and hence requires a different comparison method than single objectives. In this paper, we used two performance indicators to compare the efficiency of MOIG against NSGA-II. The first one, consisting of the number of solutions per

Algorithm 3 NSGA-II

```

Initialize population  $P_{t=0}$  with  $N$  individuals
Evaluate objective functions for each individual in  $P_{t=0}$ 
while termination criterion not met do
  Generate offspring population  $Q_t$ .
  Evaluate objective functions for each individual in  $Q_t$ 
  Merge  $P_t$  and  $Q_t$  to form  $R_t$ 
  Apply N-DS to  $R_t$  to assign individuals to fronts
  Calculate CD for each individual in each rank
   $P_{t+1} = \text{Survivor}(R_t)$ 
   $t \leftarrow t + 1$ 
end while
return the set  $P_t$ 

```

front $N\alpha$ used widely such as [37]. The second metric is the hypervolume indicator I_h proposed in [38]. It is a popular performance metric in multi-objective optimization that measures the size of the space dominated by a set of points where a higher value of I_h indicates a better frontier. I_h is also widely used in scheduling literature such as [32] and [39]. First, all values have to be normalized and then we consider a reference point by multiplying the worst objectives values by 1.2. As a result, the maximum value of I_h could be 1.44. For details of hypervolume indicator parameter computations, readers are referred to [38] and *Pygmo* library ([40]) documentation in Python that is used to compute I_h values in this study.

To test the performance of different approaches, first, all of them are adjusted and re-adapted to our problem. Secondly, a set of instances were generated by combining three parameters: the number of jobs (n), the number of stages (g), and the number of machines per stage (m). The benchmark comprises 36 instances generated using the combinations of these parameters: $n = (10, 15, 20, 30, 50, 100)$, $g = (2, 3, 4)$, and $m = (2, 3)$. The processing times (in hours) are generated by a uniform in $[1..99]$ while the energy consumption (in kWh) is generated by a uniform random in $[1..3]$, $[5..7]$, and $[3..5]$ for respectively idle, blocking, and processing energy consumption.

All procedures are coded with C++ using Visual Studio 2022 integrated development environment (IDE). All experiment tests are performed on a PC with Microsoft Windows 10 operating system, Intel Core i7 processor and 16 GB RAM.

B. Computational comparisons

The test benchmark is used to compare the MOIG performance against NSGA-II. To guarantee fairness, the same stopping criterion is used in both methods with a $TimeLimit = n * g * CPU$. The considered CPU in this paper is $CPU = 1200ms$ which returns 480 seconds for the largest instance with 100 jobs, and 4 stages with 3 identical parallel machines in each. The termination criteria computation method is adapted from [32] where similar benchmarking is provided for a multi-objective flowshop scheduling problem. Furthermore, this method allows defining a dynamic stopping criteria as a function of the instance size where shorter for small and longer

TABLE II
FRONT SIZES AND HYPER-VOLUME INDICATOR FOR INSTANCES.

<i>Ins</i>	<i>n</i>	<i>g</i>	<i>m</i>	MOIG		NSGA-II		Overlap
				N_α	I_h	N_α	I_h	
1	10	2	2	7	0.76	3	0.75	0
2	10	2	3	8	1.26	4	0.71	0
3	10	3	2	9	1.28	3	0.72	3
4	10	3	3	9	0.82	5	0.72	0
5	10	4	2	7	1.24	4	0.88	0
6	10	4	3	15	0.83	6	0.71	0
7	15	2	2	8	0.97	2	0.43	0
8	15	2	3	12	0.95	4	1.16	0
9	15	3	2	13	0.8	3	1.19	0
10	15	3	3	13	1.21	5	0.59	0
11	15	4	2	2	0.43	4	0.89	0
12	15	4	3	7	0.92	3	0.53	0
13	20	2	2	7	0.99	5	0.51	0
14	20	2	3	9	0.83	5	1.21	0
15	20	3	2	5	1.16	4	0.81	0
16	20	3	3	13	1.21	6	0.75	0
17	20	4	2	11	1.09	9	0.72	0
18	20	4	3	11	1.12	5	0.74	0
19	30	2	2	10	1.07	5	1.19	0
20	30	2	3	21	1.17	4	0.46	0
21	30	3	2	9	1.05	4	0.86	0
22	30	3	3	10	1.15	4	0.53	0
23	30	4	2	3	1	6	1	0
24	30	4	3	4	0.72	6	0.93	0
25	50	2	2	9	0.8	3	0.43	0
26	50	2	3	12	1.12	4	0.56	0
27	50	3	2	6	0.89	9	0.83	0
28	50	3	3	2	0.43	5	0.76	0
29	50	4	2	7	1.26	4	0.46	0
30	50	4	3	15	1.16	5	0.64	0
31	100	2	2	4	1.06	3	0.76	0
32	100	2	3	5	1.32	6	0.73	0
33	100	3	2	4	0.53	8	1.22	0
34	100	3	3	5	0.59	9	0.85	0
35	100	4	2	7	1.27	4	0.87	0
36	100	4	3	17	1.34	8	1.25	0
Mean				8.77	0.99	4.91	0.78	

for larger instances. Adapted termination criteria particularly effective in comparing performance of two algorithms in practical scenarios where computational resources might be a concern if algorithms are subject to run real time or in edge devices. Table II presents result of two metrics for the 36 instances that are grouped as small (instance 1-12), medium (instance 13-24) and large (instance 25-36) where best values are in bold. We notice that MOIG produces better results in most instances and for both indicators. Starting with the number of non-dominated solutions per front, this metric provides a variety of scenarios for decision-makers according to their priority. The proposed MOIG is able to give between 2 and 21 solutions for small and medium instances and between 2 and 17 solutions for large instances while NSGA-II is able to provide at most 9 different solutions. To conclude, the average N_α is 8.77 for MOIG and 4.91 for NSGA-II.

Table II shows I_h for all instances. It is clear that MOIG yields better results. For small and medium instances, NSGA-II beats the proposed MOIG only in five instances while it was better in two large instances. Therefore it can be concluded that MOIG yields better results than the NSGA-II algorithms

for both I_h and N_α benchmarking criteria for multi-objective BHFS problems with makespan and total energy consumption objectives. Finally, we also present number of overlapping solutions in both methods in the last column of Table II which indicates except instance three, solutions found are distinct.

VII. CONCLUSION AND FUTURE RESEARCH

In this paper we introduced an effective approach based on iterated greedy algorithm to tackle the multi-objective blocking hybrid flowshop scheduling problem with makespan and TEC. The problem presented here can accommodate numerous practical cases that are commonly encountered in real-world production facilities. To the best of our knowledge, this problem has never been considered in the literature. A benchmark of 36 instances comprises small, medium, and large instances are generated to test the performance of the multi-objective iterated greedy algorithm against the well-known NSGA-II adjusted for our problem. All experiment tests are done in the same environment and with the same stopping criterion. The results show that MOIG performs better than NSGA-II in all instance sizes.

Presented study can be extended in several future research directions. An exact mathematical model and solution method such as Mixed Integer Programming (MIP) and Benders decomposition exploiting that mathematical model can be considered. Constraint Programming (CP) is a proven artificial intelligence based exact method for scheduling problems that certainly give a promising research direction coupled with epsilon constraint method for adapting multi-objective nature of the problem. Developing novel heuristics such as MIP based matheuristic or CP based large neighbourhood search heuristics can be developed to benchmark with MOIG presented in this paper.

Presented work can be extended in terms of problem definitions as well. Distributed version of the BHFS is a promising direction in this regard where multiple facilities with individual BHFS systems bring an additional decision problem about which facility will be used for which set of machines. New energy-related aspects such as the machine speed, peak power, or sequence-dependent setup times between jobs could find more practical areas to implement as well.

ACKNOWLEDGMENT

This work is funded by Insight Centre for Data Analytics IoT/TU collaboration, "Models and algorithms for energy-efficient scheduling" project.

REFERENCES

- [1] E. Commission and Eurostat, *Key figures on Europe : 2021 edition*. Publications Office, 2021.
- [2] U. N. D. of Economic and S. Affairs. Sustainable development goals. [Online]. Available: <https://sdgs.un.org/goals>
- [3] M. L. Pinedo, *Scheduling Theory, Algorithms, and Systems*. Springer, 2012, ch. 1.2.
- [4] S. M. Johnson, "Optimal two- and three-stage production schedules with setup times included," *Naval Research Logistics Quarterly*, vol. 1, no. 1, pp. 61–68, 1954.

- [5] J. N. D. Gupta, "Two-stage, hybrid flowshop scheduling problem," *The Journal of the Operational Research Society*, vol. 39, no. 4, pp. 359–364, 1988.
- [6] M. F. F. Ab. Rashid, A. N. Mohd Rose, and N. M. Z. Nik Mohamed, "Hybrid flow shop scheduling with energy consumption in machine shop using moth flame optimization," in *Recent Trends in Mechatronics Towards Industry 4.0: Selected Articles from iM3F 2020, Malaysia*. Springer, 2022, pp. 77–86.
- [7] S. Wang, X. Wang, F. Chu, and J. Yu, "An energy-efficient two-stage hybrid flow shop scheduling problem in a glass production," *International Journal of Production Research*, vol. 58, no. 8, pp. 2283–2314, 2020.
- [8] J.-g. Duan, Q.-l. Zhang, Y. Zhou, and Y.-s. Wang, "Sustainable scheduling optimization of mixed-line production for large marine power components," *Journal of Cleaner Production*, vol. 280, p. 124461, 2021.
- [9] B. J. Spencer, "Pharmaceutical scheduling using simulated annealing and steepest descent method," Master's thesis, <https://doi.org/10.33915/etd.3796>, 2019.
- [10] N. G. Hall and C. Sriskandarajah, "A survey of machine scheduling problems with blocking and no-wait in process," *Operations Research*, vol. 44, no. 3, pp. 510–525, 1996.
- [11] S. Aqil and K. Allali, "Two efficient nature inspired meta-heuristics solving blocking hybrid flow shop manufacturing problem," *Engineering Applications of Artificial Intelligence*, vol. 100, p. 104196, 2021.
- [12] A. Elmi and S. Topaloglu, "A scheduling problem in blocking hybrid flow shop robotic cells with multiple robots," *Computers & operations research*, vol. 40, no. 10, pp. 2543–2555, 2013.
- [13] H.-X. Qin, Y.-Y. Han, B. Zhang, L.-L. Meng, Y.-P. Liu, Q.-K. Pan, and D.-W. Gong, "An improved iterated greedy algorithm for the energy-efficient blocking hybrid flow shop scheduling problem," *Swarm and Evolutionary Computation*, vol. 69, p. 100992, 2022.
- [14] M. Marichelvam, M. Geetha, and Ö. Tosun, "An improved particle swarm optimization algorithm to solve hybrid flowshop scheduling problems with the effect of human factors—a case study," *Computers & Operations Research*, vol. 114, p. 104812, 2020.
- [15] D. Lei, L. Gao, and Y. Zheng, "A novel teaching-learning-based optimization algorithm for energy-efficient scheduling in hybrid flow shop," *IEEE Transactions on Engineering Management*, vol. 65, no. 2, pp. 330–340, 2017.
- [16] S. Schulz, J. S. Neufeld, and U. Buscher, "A multi-objective iterated local search algorithm for comprehensive energy-aware hybrid flow shop scheduling," *Journal of cleaner production*, vol. 224, pp. 421–434, 2019.
- [17] E.-K. Boukas, A. Haurie, and F. Soumis, "Hierarchical approach to steel production scheduling under a global energy constraint," *Annals of operations research*, vol. 26, no. 1–4, pp. 289–311, 1990.
- [18] A. T. Lora, J. C. Riquelme, J. L. M. Ramos, J. M. R. Santos, and A. G. Expósito, "Application of evolutionary computation techniques to the optimal short-term scheduling of the electrical energy production," in *CAEPIA*. Springer, 2003, pp. 656–665.
- [19] A. A. Bruzzone, D. Anghinolfi, M. Paolucci, and F. Tonelli, "Energy-aware scheduling for improving manufacturing process sustainability: A mathematical model for flexible flow shops," *CIRP annals*, vol. 61, no. 1, pp. 459–462, 2012.
- [20] H. Luo, B. Du, G. Q. Huang, H. Chen, and X. Li, "Hybrid flow shop scheduling considering machine electricity consumption cost," *International journal of production economics*, vol. 146, no. 2, pp. 423–439, 2013.
- [21] F. Keller, C. Schönborn, and G. Reinhart, "Energy-orientated machine scheduling for hybrid flow shops," *Procedia Cirp*, vol. 29, pp. 156–161, 2015.
- [22] J. Long, Z. Zheng, X. Gao, and P. M. Pardalos, "A hybrid multi-objective evolutionary algorithm based on nsga-ii for practical scheduling with release times in steel plants," *Journal of the Operational Research Society*, vol. 67, no. 9, pp. 1184–1199, 2016.
- [23] J.-q. Li, H.-y. Sang, Y.-y. Han, C.-g. Wang, and K.-z. Gao, "Efficient multi-objective optimization algorithm for hybrid flow shop scheduling problems with setup energy consumptions," *Journal of Cleaner Production*, vol. 181, pp. 584–598, 2018.
- [24] K. Fang, N. A. Uhan, F. Zhao, and J. W. Sutherland, "Flow shop scheduling with peak power consumption constraints," *Annals of Operations Research*, vol. 206, pp. 115–145, 2013.
- [25] D. Kizilay, M. F. Tasgetiren, Q.-K. Pan, and G. Süer, "An ensemble of meta-heuristics for the energy-efficient blocking flowshop scheduling problem," *Procedia Manufacturing*, vol. 39, pp. 1177–1184, 2019.
- [26] X. Zhang, X. Liu, A. Cichon, G. Królczyk, and Z. Li, "Scheduling of energy-efficient distributed blocking flowshop using pareto-based estimation of distribution algorithm," *Expert Systems with Applications*, vol. 200, p. 116910, 2022.
- [27] S.-L. Jiang and L. Zhang, "Energy-oriented scheduling for hybrid flow shop with limited buffers through efficient multi-objective optimization," *IEEE Access*, vol. 7, pp. 34 477–34 487, 2019.
- [28] J. M. Framinan and R. Leisten, "A multi-objective iterated greedy search for flowshop scheduling with makespan and flowtime criteria," *OR Spectrum*, vol. 30, pp. 787–804, 2008.
- [29] J.-Y. Ding, S. Song, and C. Wu, "Carbon-efficient scheduling of flow shops by multi-objective optimization," *European Journal of Operational Research*, vol. 248, no. 3, pp. 758–771, 2016.
- [30] H.-x. Qin, Y. Han, J. Li, H. Sang, Q. Chen, L. Meng, and B. Zhang, "A quick and effective iterated greedy algorithm for energy-efficient hybrid flow shop scheduling problem with blocking constraint," in *2021 11th International Conference on Information Science and Technology (ICIST)*. IEEE, 2021, pp. 325–331.
- [31] R. Ruiz and T. Stützle, "A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem," *European journal of operational research*, vol. 177, no. 3, pp. 2033–2049, 2007.
- [32] G. Minella, R. Ruiz, and M. Ciavotta, "Restarted iterated pareto greedy algorithm for multi-objective flowshop scheduling problems," *Computers & Operations Research*, vol. 38, no. 11, pp. 1521–1533, 2011.
- [33] M. Nawaz, E. E. Ensore Jr, and I. Ham, "A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem," *Omega*, vol. 11, no. 1, pp. 91–95, 1983.
- [34] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.
- [35] M. Babor, L. Pedersen, U. Kidmose, O. Paquet-Durand, and B. Hitzmann, "Application of non-dominated sorting genetic algorithm (nsga-ii) to increase the efficiency of bakery production: A case study," *Processes*, vol. 10, no. 8, p. 1623, 2022.
- [36] Q. Luo, Q. Deng, G. Xie, and G. Gong, "A pareto-based two-stage evolutionary algorithm for flexible job shop scheduling problem with worker cooperation flexibility," *Robotics and Computer-Integrated Manufacturing*, vol. 82, p. 102534, 2023.
- [37] A. Mehrabian, R. Tavakkoli-Moghaddam, and K. Khalili-Damaghani, "Multi-objective routing and scheduling in flexible manufacturing systems under uncertainty," *Iranian journal of fuzzy systems*, vol. 14, no. 2, pp. 45–77, 2017.
- [38] E. Zitzler, J. Knowles, and L. Thiele, "Quality assessment of pareto set approximations," *Multiobjective optimization: Interactive and evolutionary approaches*, pp. 373–404, 2008.
- [39] X. Li and S. Ma, "Multi-objective memetic search algorithm for multi-objective permutation flow shop scheduling problem," *IEEE access*, vol. 4, pp. 2154–2165, 2016.
- [40] F. Biscani *et al.*, "PyGMO: Python Parallel Global Multiobjective Optimizer," 2021. [Online]. Available: <https://github.com/esa/pagmo/tree/devel/pygmo>