

Title	Multi-objective energy-efficient scheduling in two-stage hybrid flowshop under consideration of no-wait
Authors	Missaoui, Ahmed;O'Sullivan, Barry
Publication date	2025-07-12
Original Citation	Missaoui, A. and O'Sullivan, B. (2025) 'Multi-objective energy-efficient scheduling in two-stage hybrid flowshop under consideration of no-wait', in Fujita, H., Watanobe, Y., Ali, M. and Wang, Y. (eds) Advances and Trends in Artificial Intelligence. Theory and Applications. IEA/AIE 2025. Lecture Notes in Computer Science,15707, pp. 464-475. Springer, Singapore. https://doi.org/10.1007/978-981-96-8892-0_39
Type of publication	Article (peer-reviewed);book-chapter;Conference item
Link to publisher's version	10.1007/978-981-96-8892-0_39
Rights	© 2025, the Authors, under exclusive license to Springer Nature Singapore Pte Ltd. This version of the paper has been accepted for publication, after peer review (when applicable) and is subject to Springer Nature's AM terms of use, but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record is available online at: https://doi.org/10.1007/978-981-96-8892-0_39
Download date	2026-10-23 21:11:51
Item downloaded from	https://hdl.handle.net/10468/17873

Multi-Objective Energy-Efficient Scheduling in Two-Stage Hybrid Flowshop under Consideration of No-wait

Ahmed Missaoui¹ and Barry O’Sullivan¹

Insight SFI Centre for Data Analytics, University College Cork, Ireland
ahmed.missaoui@insight-centre.org, b.osullivan@cs.ucc.ie

Abstract. The industrial sector is one of the world’s largest energy consumers. Moreover, the fluctuating energy costs make effective energy management a critical challenge for the manufacturing industry. To address these challenges, companies are increasingly focusing on optimizing energy-efficient scheduling practices. In the current work, we addressed the two-stage no-wait hybrid flowshop problem, aiming to minimize the total completion time and energy consumption. We initially introduced a mixed integer linear programming (MILP) model when the augmented epsilon-constraint is employed to generate the optimal Pareto front for small instances. In the second step, an efficient bi-objective iterated local search algorithm is introduced to solve a benchmark of 100 instances. The results obtained are compared against those from the Non-dominated Sorting Genetic Algorithm. The comparative analysis demonstrates our proposal’s superior effectiveness.

Keywords: Hybrid flowshop · No-wait · Iterated Local search · Multi-objective optimization · Makespan · Energy consumption.

1 Introduction

Due to the increased production demands, the manufacturing sector now accounts for over a quarter of total energy consumption. Consequently, it is crucial to adopt new strategies that are able to reduce energy consumption. In order to achieve this objective, industrial companies have two intuitive options. In first hand, it is possible to invest in new technologies as long-term strategies that require significant resources. In second hand, companies can manage current resources more efficiently as short-term strategies, offering a practical solution that can significantly reduce energy costs. In literature, researchers and academics refer to this approach as energy-efficient scheduling (EES). EES is a set of actions and setups designed to reduce energy waste resulting from unnecessary operations like machine standby and blocking time [1]. As a result, EES has progressively attracted interest, emerging as a crucial approach able to enhance energy efficiency and lower carbon emissions.

In this research work, we addressed the Energy-Efficient Two-Stage Hybrid Flowshop with no-wait Constraints (EE-HFS-NW), aiming to minimize both

the total completion time and energy consumption (TEC). However, minimizing Makespan helps to decrease production pressure and increase resource utilization, while minimizing TEC reduces energy costs and improves sustainability.

Properly, the Hybrid Flowshop Scheduling problem (HFS) aims to effectively schedule and process a set of n jobs across a set of g stages when each of them involves m identical parallel machines. However, in this work, the HFS problem with two-stage is addressed where the first stage operates with a single machine, while the second one is equipped with m parallel machines. Each job should be processed in the first stage and then immediately moved to one of the available machines in the second stage, where the processing time is known in advance. Compared with flowshop problems, HFS structure leads to more complexity, as it involves both sequencing and allocating of jobs. Refer to [2], the HFS issue is NP-hard even in the case of two stages, when the first one includes one machine and the second stage involves two machines.

Regardless of its complexity, the HFS problem is still of great importance in several manufacturing industries such as automobile manufacturing, healthcare management and, food production. The diversity of application fields emphasizes the importance of HFS production system in manufacturing.

In a no-wait environment, each job must be processed uninterruptedly from beginning to end during transitions between production stages. Nowadays, No-wait has emerged in many manufacturing systems that deal with perishable goods, such as nutritional and chemical products.

In the literature, several multi-objective approaches have been introduced to deal with scheduling problems effectively. During this work, a mathematical model is introduced for the exact resolution of the studied problem, while an effective bi-objective iterated local search (MOILS) is designed to give the Pareto fronts of the two considered objectives. Readers can refer to [3] for more resolution approaches and energy aspects of EES.

To our knowledge, our work is the first that explore the bi-objective EE-HFS-NW, focusing on minimizing both TEC and Makespan simultaneously. This paper is structured as: Section 2 details such related works. Section 3 presents the developed mathematical model. The proposed resolution approaches are explained in detail in Section 4. Experimental setups and computational results are discussed in Section 5. A summary of our work and some suggestions for future research are provided in Section 6.

2 Related works

The approach in [4] is one of the earliest works that investigated EES. In this work, the authors studied the HFS problem in steel industries in order to minimize Makespan while taking into account the power constraints.

Since then, there has been considerable growth in publications investigating EES in HFS problems using several resolution approaches. The HFS problem with transportation and setup times is studied in [5] and solved by applying a

multi-objective approach based on decomposition. The distributed HFS is addressed in [6] and solved using a bi-objective iterated greedy, with the objectives being makespan and TEC. In [7], the authors study a two-stage HFS with time-of-use (TOU) electricity prices, aiming to minimize makespan and energy costs. A MILP model is formulated, and the augmented ϵ -constraint is applied to obtain the exact Pareto front for small benchmark instances. Finally, a set of metaheuristics is used for approximate resolution. Recently, the HFS with blocking constraints is talked in [1], where the objective is to reduce Makespan and TEC using an approach based on multi-objective iterated greedy (MOIG). Despite its importance and relevance to many specific manufacturing processes, the no-wait constraint is rarely studied in terms of energy efficiency. However, back to [3], this constraint is addressed in only five papers. In [8], the distributed no-wait flowshop scheduling problem is tackled using reinforcement learning-based metaheuristics. As can be seen, population-based methods are widely used for energy-aware scheduling problems. However, iterated approaches such as MOILS and MOIG have also proven effective for this type of multi-objective problem. In particular, MOILS offers a flexible and efficient alternative to handle multi-objective optimization by integrating suitable selection mechanisms to balance intensification and diversification. As far as we know, the energy-aware HFS with no-wait constraints has never been studied before.

3 Statement of the Problem

3.1 MILP model details

This section gives the MILP model representation for the two-stage no-wait HFS problem inspired from [7].

Parameters:

- n : jobs indexed by j, k .
- $g = 2$: the two stages are indexed by i
- m_i : machines per stage i indexed by l .
- p_{ji} : processing time of the job j in stage i .
- f_h : set of speeds indexed by h .
- $U_h = f_h \times f_h$
- EO_h : the power consumption of machine l at speed h
- Ide_i : idle time energy consumption at stage i .
- M : Large integer.

Decision variables:

- $Cmax$: Makespan
- TEC : Total energy consumption
- B_{ji} : Starting time of job j at stage i
- C_{ji} : Completion time of job j at stage i
- S_{ij} : speed level of job j at stage i .
- fp_{ij} : Processing time considering the speed levels for the job j at stage i
- $dl_{i,l}$: total idle time of machine l at stage i

dl_i : total idle time in stage i

$MC_{l,i}$: completion time of machine l at stage i

tpe : total processing energy consumption

fp_i : Sum of processing time considering the speed levels at stage i

$X_{jli} = 1$, if job j is assigned to machine l at stage i and 0, otherwise.

$Z_{jki} = 1$, if job j precedes job k at stage i and 0, alternatively.

$V_{jih} = 1$, if job j is handled with speed h at stage i and 0 in contrast.

MILP model description:

$$\text{Min } C_{max} \quad (1)$$

$$\text{Min } TEC \quad (2)$$

$$\sum_{l=1}^{m_i} X_{jli} = 1, \forall j \in \{1..n\}, \forall i \in \{1..g\} \quad (3)$$

$$Z_{j,k,i} + Z_{k,j,i} \leq 1, \forall j, k \in \{1..n\}, j \neq k, \forall i \in \{1..g\} \quad (4)$$

$$Z_{j,k,i} + Z_{k,j,i} \geq X_{jli} + X_{kli} - 1, \forall j, k \in \{1..n\}, j \neq k, \forall i \in \{1..g\} \quad (5)$$

$$B_{k,i} - C_{j,i} + M(3 - X_{jli} - X_{kli} - Z_{j,k,i}) \geq 0, \forall k, j \in \{1..n\}, j \neq k, \forall i \in \{1..g\} \quad (6)$$

$$\sum_{h \in f} V_{jih} = 1, \forall j \in \{1..n\}, \forall i \in \{1..g\} \quad (7)$$

$$fp_{ji} = \sum_{h \in f} (V_{jih} \times p_{ji} \div f_h), \forall j \in \{1..n\}, \forall i \in \{1..g\} \quad (8)$$

$$C_{j,i} = B_{j,i+1}, \forall i \in \{1..n\}, \forall i \in \{1..g-1\} \quad (9)$$

$$C_{ji} = B_{ji} + fp_{ji}, \forall j \in \{1..n\}, \forall i \in \{1..g\} \quad (10)$$

$$C_{j,i} \leq MC_{li} + M(1 - X_{jli}), \forall j \in \{1..n\}, \forall l \in \{1..m_j\}, \forall i \in \{1..g\} \quad (11)$$

$$dl_{i,l} = MC_{li} - \sum_{j=1}^n (fp_{ji} * X_{jli}), \forall j \in \{1..n\}, \forall l \in \{1..m_i\}, \forall i \in \{1..g\} \quad (12)$$

$$dl_i = \sum_{l \in m_i} dl_{i,l} \forall i \in \{1..g\} \quad (13)$$

$$fp_i = \sum_{j \in n} fp_{ji}, \forall i \in \{1..2\} \quad (14)$$

$$tpe = \sum_{i \in g} \sum_{j \in n} \sum_{h \in f} (p_{ji} \div f_h) \times V_{jih} \times U_h \times 4 \quad (15)$$

$$C_{max} \geq C_{j,g-1}, \forall j \in \{1..n\} \quad (16)$$

$$TEC = tpe + \sum_{i \in g} (dl_i \times Ide_i) \quad (17)$$

(1) and (2) refer to the Makespan and TEC objective functions, respectively. Constraints (3) assure that each job is handled by one machine per station. Constraints (4) ensure that at the two stages and for any two jobs, only one can precede the other. Constraints (5) ensure that if two jobs are handled on the same machine, one must precede the other. Constraints (6) prevent overlapping of jobs assigned to the same machine at the same stage. Constraints (7) ensure that each job is handled at just one speed level. Constraints (8) calculate the real processing time of each job at all stages. Constraints (9) assure the no wait constraints. Constraints (10) to relate the starting time and completion time of a given job. Constraint (11) identifies the shutdown time for any machines across the two stages. Constraints (12) and Constraints (13) calculate the idle time per machine and per stage. Constraints (14) compute the sum of processing time in each stage. Constraints (15) compute the energy consumption during processing as defined in the work of [9]. Finally, Constraints (16) and (17) are used to determine the two objective functions respectively.

Several methods are proposed in the literature for solving multi-objective mathematical models, including the lexicographic method and the ϵ -constraint approach. In this work, the augmented ϵ -constraint approach is used to acquire Pareto fronts.

3.2 Illustrative Example

Job	Processing Times ($p_{j,i}$)		Speed Levels	
	i = 1	i = 2	i = 1	i = 2
1	3	12	1	3
2	4	10	2	2
3	2	12	1	3
4	6	6	2	3
5	6	12	3	3
6	9	2	3	1

Table 1. Processing times and speed levels

To explain the problem better, we present an example of a simple instance including 6 jobs and 3 machines at the second stage to obtain the total completion time and TEC. Table 3.2 presents the processing times as well as speed levels for both stages. The idle energy consumption is 3 and 2 respectively for stages 1 and 2. The resulting Gantt chart of $\pi = (1, 2, 3, 4, 5, 6)$ is shown in Fig. 1. The above example reveals a $Cmax$ of 17 and 818 as TEC.

4 Multi-objective Iterated Local Search

In the scheduling literature, several variations of Multi-Objective Iterated Local Search (MOILS) have been suggested, proving its effectiveness in solving

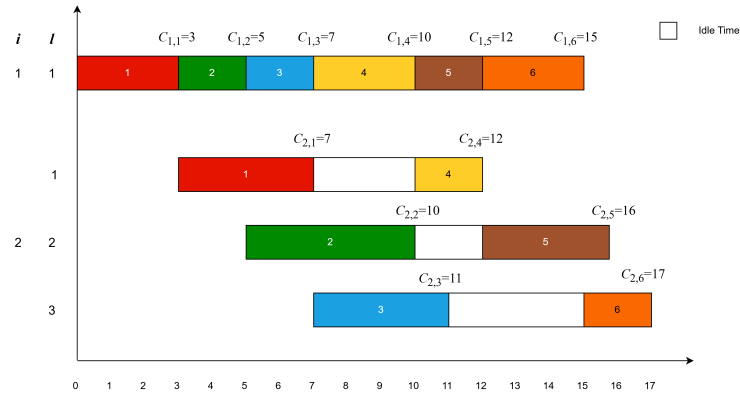


Fig. 1. Gantt chart of the EE- HFS-NW example

scheduling problems, which served as the motivation for our choice. The key phases are outlined in Fig 2. In the following sections a detailed explanation is provided for each step.

4.1 Initial solution

The effectiveness of the preliminary solution has an effect on the effectiveness of any resolution method. In this work, one solution is generated for each objective. Starting with Makespan, the NEH heuristic from [10] is applied when jobs are processed with the highest speed levels. For energy consumption, we used the NEH heuristic adopted in [9]. As a result, the two initial solutions formed the first Pareto front.

4.2 Selection operator

Moving from one operator to another, an ensemble of non-dominated solutions is generated. The concept of handling all solutions at each algorithm phase doesn't look promising as it can be resource-intensive. The solution here is to select at each time one solution to be handled in the next step. This selection is revealed on the Crowding Distance (CD) introduced in [11], where the solution with the maximum CD is chosen.

4.3 Perturbation

For a selected solution, perturbations are applied to both the job sequence and the speed levels. In the first step, one random job is chosen and reinserted in all positions without modification of speed levels. In the second step, a random change is applied to the speed level of each job, with each modification producing a new solution, we have to mention that the previous speed level changes are kept

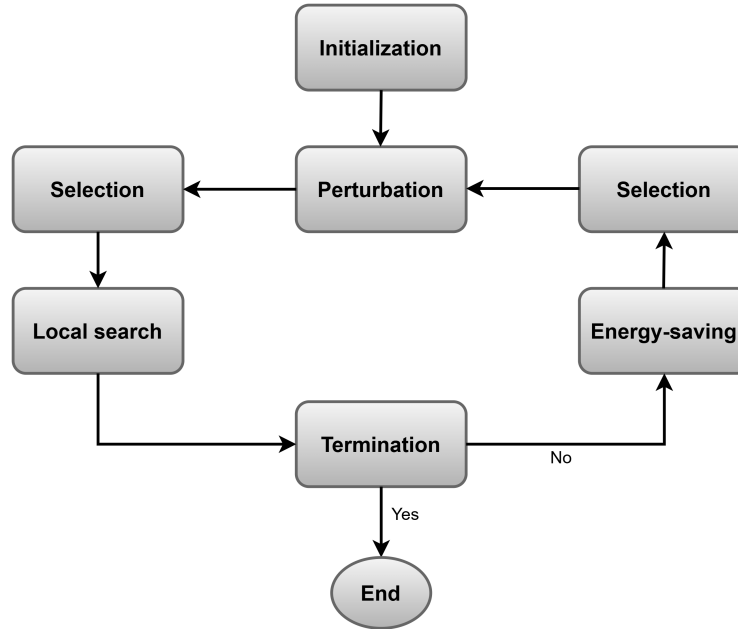


Fig. 2. Flowchart of MOILS.

each time. The two resulting lists of solutions are then combined with the current Pareto front. The non-dominated sorting algorithm is applied leaving only the non-dominated solutions. For better clarity, all the details of the perturbation phase are provided in Algorithm 1.

4.4 Local search

This operator uses two neighborhood structures based on insertion and exchange. The first structure involves randomly selecting two jobs, exchanging their position, and applying a random modification to their speed levels, repeating this process up to k times. The second structure selects a single random job and inserts it into all positions, each time applying a random modification to its speed levels. The resulting list of all new solutions is then combined with the existing Pareto front and only the non-dominated solutions are retained to create the updated Pareto front. The main steps of local search are given in algorithm 2.

4.5 Energy-saving strategy

To reduce TEC without affecting Makespan, adjustments can be made during idle periods between the completion of one job and the start of the next on the same machine. If the machine's speed level is not already at its minimum, the speed is gradually decreased to the lowest level. This adjustment could increase

Algorithm 1 Perturbation

Input: π_1
Output: S
 Select one random job j
for $i := 1$ to n **do**
 $\pi \leftarrow$ Insert the job j in position i
 Add π to B
end for
for $i := 1$ to n **do**
 $\pi \leftarrow$ random change in speed levels of job i
 Add π to B
end for
 $S \leftarrow$ non dominated solution of B

Algorithm 2 Local Search Procedure

1: **Input:** Initial solution π_1
 2: **Output:** Set of solutions S
 3: Initialize iteration counter $k \leftarrow 0$
 4: **while** $i < k$ **do**
 5: Select a random job j
 6: Select a random neighborhood structure n_c
 7: **if** $n_c = 1$ **then**
 8: $\pi \leftarrow \text{exchange}(\pi)$
 9: Add π to S
 10: **else if** $n_c = 2$ **then**
 11: $\pi \leftarrow \text{insertion}(\pi)$
 12: Add π to S
 13: **end if**
 14: $i \leftarrow i + 1$
 15: **end while**
 16: $S \leftarrow$ non-dominated solutions in B

the processing time of the job but help reduce energy consumption without affecting Makespan. The process is repeated iteratively, moving to the next lower speed level whenever possible. This operation continues whenever idle time exists on all machines. This procedure is applied to all solutions on the Pareto front.

5 Computational Assessment and Statistical Testing

5.1 Experimental Setup

A benchmark with different sizes is generated to evaluate the performance of the proposed methods. The instances correspond to combinations of two parameters: n in the range (10, 20, 30, 50) and the number of resources in the second stage m_1 in the range (2, 3, 5, 8, 10). The values of the standard processing times are randomly generated from a uniform distribution $U[1..99]$, while the idle energy consumption per time unit is drawn from a uniform distribution $U[1..3]$. As used in [9], the speed levels are set as $f_h = (1.00, 1.30, 1.55, 1.75, 2.10)$. The benchmark includes 5 replications for each combination of parameters, resulting in a total of 100 instances.

To evaluate the effectiveness of the proposed MOILS, we used as a competing method the NSGA-II introduced by [11]. The crossover phase is performed using Partial-Mapped Crossover (PMX), while mutation is realized through exchanges.

For small instances comprising 10 jobs, the performance of our MOILS is evaluated against NSGA-II and the augmented epsilon-constraints method. Two metrics are employed to assess the effectiveness of the three approaches. The Hyper-volume Indicator I_h which measures the area of the Pareto front according to the reference point of (1.0, 1.0). The Generational Distance GD measures the average distance between the obtained solutions and the global Pareto front.

To ensure greater effectiveness of the methods used, a calibration step is performed. The proposed MOILS requires only one parameter: the number of iterations in the local search k . To determine the optimal value of k , we applied the design of experiments (DOE) method to 20 randomly generated instances with different parameter combinations. The parameter k was tested at three levels (100, 200, 300). The DOE analysis indicates that $k = 100$ is the best value. For NSGA-II, the parameter settings proposed by [12] are used.

The proposed MOILS as well as the NSGA-II are implemented in C++ within the Microsoft Visual Studio 22. For the exact approach, the IBM Cplex solver is used to implement the MILP model. All experiments took place on a Linux server with an Intel(R) Xeon(R) processor and 32 GB of RAM. For each method, a stopping criterion of $CPU = (n \times g \times 200)$ milliseconds is applied for every instance. For the MILP model and the augmented ϵ -constraint method, a total stopping time of one hour is allocated for 20 grids, with each grid running for 3 minutes.

5.2 Computational Results

Once the experimental tests are finished, the computational results are used to evaluate the performance of the introduced methods. For small instances with

10 jobs, the proposed MOILS is compared against NSGA-II and the augmented ϵ -constraints approach (Ep). A comprehensive summary of the results can be found in Table 2. It is clear that *Ep* performs better, with an average I_h of 0.73 and an average GD equal to 0.109. The *Ep* approach produces the best results, while MOILS achieves the closest results in both criteria.

Table 2. I_h and GD values for small instances

Jobs	Machines	I_h			GD		
		MOILS	NSGA	Ep	MOILS	NSGA2	Ep
10	2	0.71	0.68	0.79	0.114	0.146	0.099
10	3	0.78	0.69	0.83	0.141	0.178	0.112
10	5	0.75	0.63	0.82	0.155	0.192	0.124
10	8	0.76	0.62	0.79	0.128	0.167	0.114
10	10	0.69	0.58	0.74	0.134	0.168	0.098
Average		0.73	0.64	0.79	0.134	0.170	0.109

In the second step, all benchmark instances are considered to assess the performance of the MOILS approach against NSGA-II. All results are structured by n and m_1 are presented in Table 3.

As can be seen, the differences in I_h are not particularly significant, while MOILS did not outperform NSGA-II in all instances. In other hand, it is clear that MOILS is able to give better results for small instances of 10 jobs. In general, our proposed MOILS shows a considerable advantage over NSGA-II, achieving an average I_h of 0.75, which represents an improvement of 10.29% compared to the NSGA-II I_h . It is clear that the results for the GD metric are related to I_h ; however, MOILS doesn’t outperform NSGA-II in all instance sizes. As for the first metric, MOILS is better than NSGA-II for all small instances with 10 jobs. In general, MOILS accomplish an average GD of 0.138, while NSGA-II produces an average GD of 0.168. Figure 3 shows the two obtained Pareto fronts of an instance of 30 jobs.

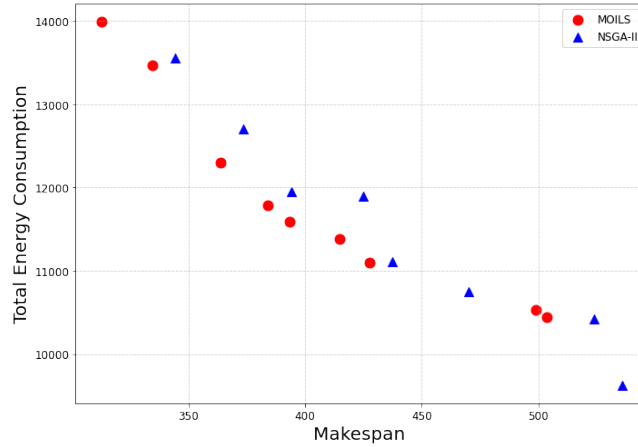
6 Conclusion

In our research work, we coped with the problem of two-stage hybrid flowshop scheduling with no-wait constraints. We introduced a mixed-integer linear programming model alongside a multi-objective iterated local search approach, considering both makespan and total energy consumption. A benchmark of 100 instances was used to evaluate the effectiveness of the MOILS against the NSGA-II, and the MILP models. Our findings reveal that while the augmented ϵ -constraint approach performs best for small instances while the proposed MOILS achieves superior results for larger instances.

Studying the same problem in its general form or by developing a distributed version presents a promising direction for future research. Furthermore, consider-

Table 3. I_h and GD values for all instances

Jobs	Machines	I_h		GD	
		MOILS	NSGA	MOILS	NSGA2
n	m	ILS	NSGA	ILS	NSGA2
10	2	0.77	0.70	0.111	0.142
10	3	0.86	0.74	0.135	0.174
10	5	0.83	0.63	0.141	0.186
10	8	0.82	0.75	0.124	0.145
10	10	0.73	0.67	0.119	0.158
20	2	0.83	0.58	0.127	0.159
20	3	0.68	0.70	0.114	0.110
20	5	0.71	0.74	0.231	0.245
20	8	0.73	0.67	0.148	0.215
20	10	0.76	0.64	0.133	0.158
30	2	0.73	0.74	0.107	0.124
30	3	0.75	0.70	0.137	0.158
30	5	0.71	0.73	0.201	0.255
30	8	0.69	0.65	0.118	0.231
30	10	0.73	0.70	0.115	0.141
50	2	0.79	0.81	0.138	0.132
50	3	0.73	0.62	0.129	0.138
50	5	0.74	0.75	0.115	0.111
50	8	0.77	0.53	0.214	0.256
50	10	0.72	0.73	0.117	0.129
Average		0.75	0.68	0.138	0.168

**Fig. 3.** Pareto front of MOILS and NSGA-II.

ing additional practical situations for both energy consumption and productivity looks interesting.

Acknowledgments. This publication was conducted with the financial support of Science Foundation Ireland under Grant number 12/RC/2289-P2 at Insight the SFI Research Centre for Data Analytics at UCC and 22/NCF/DR/11264, National Challenge Fund, Digital for Resilience Challenge.

References

1. A. Missaoui, C. Ozturk, B. O’Sullivan, Energy-efficient multi-objective hybrid flow-shop scheduling problem with blocking constraints, in: 2023 9th International Conference on Control, Decision and Information Technologies (CoDIT), IEEE, 2023, pp. 01–06.
2. J. N. D. Gupta, Two-stage, hybrid flowshop scheduling problem, *The Journal of the Operational Research Society* 39 (4) (1988) 359–364.
3. A. Missaoui, C. Ozturk, B. O’Sullivan, M. Garraffa, Energy-efficient manufacturing scheduling: A systematic literature review, *Handbook of Formal Optimization* (2023) 1–26.
4. E.-K. Boukas, A. Haurie, F. Soumis, Hierarchical approach to steel production scheduling under a global energy constraint, *Annals of operations research* 26 (1–4) (1990) 289–311.
5. B. Zhang, Q.-k. Pan, L. Gao, L.-L. Meng, X.-Y. Li, K.-K. Peng, A three-stage multiobjective approach based on decomposition for an energy-efficient hybrid flow shop scheduling problem, *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 50 (12) (2019) 4984–4999.
6. C. Lu, Q. Liu, B. Zhang, L. Yin, A pareto-based hybrid iterated greedy algorithm for energy-efficient scheduling of distributed hybrid flowshop, *Expert Systems with Applications* 204 (2022) 117555.
7. S. Wang, X. Wang, F. Chu, J. Yu, An energy-efficient two-stage hybrid flow shop scheduling problem in a glass production, *International Journal of Production Research* 58 (8) (2020) 2283–2314.
8. F. Zhao, X. Hu, L. Wang, T. Xu, N. Zhu, Jonrinaldi, A reinforcement learning-driven brain storm optimisation algorithm for multi-objective energy-efficient distributed assembly no-wait flow shop scheduling problem, *International Journal of Production Research* 61 (9) (2023) 2854–2872.
9. A. Missaoui, C. Ozturk, B. O’Sullivan, Energy-aware blocking hybrid flow shop scheduling problem with sequence-depend setup times and machine speed levels, *Procedia Computer Science* 253 (2025) 1134–1143.
10. M. Nawaz, E. E. Ensore Jr, I. Ham, A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem, *Omega* 11 (1) (1983) 91–95.
11. K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: Nsga-ii, *IEEE transactions on evolutionary computation* 6 (2) (2002) 182–197.
12. H. Luo, B. Du, G. Q. Huang, H. Chen, X. Li, Hybrid flow shop scheduling considering machine electricity consumption cost, *International journal of production economics* 146 (2) (2013) 423–439.