

Permutation Flow Shop Scheduling Optimization with Batch Heat Treatment for Gear Manufacturing

^{1st} Bixian Song¹, ^{2nd} Lingchong Zhong¹, ^{3rd} Xing Guo², ^{4th} Zaixing Sun³, ^{5th} Xun Yong Yang², ^{6th} Hui Liu^{1*}
¹Gannan University of Science and Technology, Ganzhou, China ³Pengcheng Laboratory, Shenzhen, China

²Ganzhou Qunxing Machinery Co., Ltd, Ganzhou, China *Corresponding author. Email: 9320020058@gnust.edu.cn

Abstract—In gear manufacturing, workpieces pass through forging, turning, hobbing, carburizing and quenching, and finish grinding. The carburizing furnace is a batch processing machine (BPM) with fixed cycle time, intrinsically coupling discrete stages upstream and downstream. This paper studies the permutation-constrained flow-shop scheduling problem with embedded batch processing (PFSP-BPM) to isolate the sequencing–batching coupling. A mixed-integer linear programming (MILP) model is developed to jointly minimize makespan and total carbon emissions; explicit linearization renders it directly solvable by commercial solvers. Correctness is verified via exhaustive enumeration on 16 instances and ϵ -constraint Pareto-front analysis. Experiments reveal a parameter-dependent Pareto trade-off, while exact optimality becomes difficult to certify beyond $n = 8$. Decoupled-baseline results further show that permutation and batching decisions are structurally non-separable: optimizing either in isolation forfeits synergy gains of 4.7%–7.7%.

Index Terms—permutation flow shop, batch processing machine, mixed-integer linear programming, carbon emissions, multi-objective optimization, gear manufacturing

I. INTRODUCTION

In gear manufacturing, forged blanks proceed sequentially through forging, turning, hobbing, carburizing and quenching, and finish grinding. The carburizing furnace processes multiple gears simultaneously at a fixed cycle time, acting as a batch processing machine (BPM) that intrinsically couples upstream and downstream discrete stages: workpieces queue to form batches before the furnace and proceed downstream only upon batch completion. Heat-treatment energy accounts for 30%–50% of total gear manufacturing energy [1], making carbon emissions an important criterion alongside makespan. The discrete-batch-discrete structure also arises in semiconductor diffusion furnaces, pharmaceutical sterilization autoclaves, and ceramic sintering kilns. In dedicated heat-treatment lines, operational constraints such as fixture pallets, conveyor pacing, and traceability requirements tend to preserve job order across stages, rendering the permutation-constrained flow shop a practically relevant framework. This study takes gear ring manufacturing as a representative case.

Beyond capacity limits, furnace loads are governed by metallurgical compatibility rules: gears with comparable carbon and alloying-element content may be co-loaded, carburizing case-depth tolerance is typically around 0.3 mm, and parts

in the same furnace must be structurally similar and close in weight to ensure uniform heating. These rules are commonly enforced through upstream process planning, which pre-screens jobs into compatible families before they reach the shop floor. This paper treats such compatibility as a pre-processing condition and focuses on the resulting scheduling subproblem, reducing it to a permutation flow shop with finite-capacity batch processing in which any subset of jobs up to capacity C may be co-processed.

The Permutation Flow Shop Scheduling Problem (PFSP) has been studied extensively since Johnson’s seminal work [2] and systematically surveyed by Ruiz and Maroto [3]. Green manufacturing imperatives have spurred energy- and carbon-aware variants [1], [4], [5], and the field has expanded to distributed settings [6]. Yet the interaction between permutation constraints and batch processing machines—particularly its structural implications for exact optimization—has received limited attention.

BPM scheduling has an equally deep tradition, with foundational complexity results by Uzsoy [7] and a systematic classification by Potts and Kovalyov [8]. Recent flow-shop BPM research has concentrated on hybrid flow shop (HFS) variants, where parallel machines at each stage and sequence-independent routing relax the permutation requirement [9]–[12]. Consequently, the defining PFSP feature—a single permutation across all stages—is absent in these studies. Fowler and Mönnch [13] identified the mid-line BPM configuration as an open direction without addressing permutation constraints. Incompatible job families have been studied for standalone BPMs [14], [15] but not under flow-shop embedding with permutation constraints. Carbon-aware scheduling research [16] further treats energy as a constant carbon proxy, ignoring the occupancy-dependent efficiency of BPMs. In summary, existing work has addressed routing–batching trade-offs in HFS-BPM, incompatible families in standalone BPMs, and carbon-aware objectives in PFSP, yet the structural implications of enforcing a single permutation across all stages, including the BPM, remain to be fully characterized.

This paper makes the following contributions:

- 1) **Problem formulation and coupling characterization.** We formalize the PFSP-BPM sequencing–batching coupling through a MILP model comprising 17 constraint families. Eliminating routing flexibility concentrates the combinatorial complexity into the sequencing–batching

This work was supported in part by Gannan University of Science and Technology, Ganzhou, Jiangxi, China, under Grant 6222226029/201; and in part by the Major Key Project of PCL, China under Grant PCL2025A13.

interaction, which we characterize through exact methods. The model is made directly solvable through explicit linearization of permutation-variable products and the TCE product term;

- 2) **Coupling-strength characterization.** We show that the sequencing–batching coupling is structurally nontrivial: optimizing either decision in isolation can forfeit synergy gains of 4.7%–7.7%, established via exact decoupled-baseline comparison on small-scale instances. This finding is supported by exhaustive enumeration (16 instances, deviations $< 10^{-13}$) and ε -constraint Pareto-front verification (10^{-4} tolerance);
- 3) **Problem-structure insights.** A parameter-dependent Pareto trade-off exists between makespan and carbon emissions; under the tested formulation and solver settings, exact solution time grows exponentially with a tractability transition around $n = 8$, and no incumbent is obtained for $n \geq 12$ within one hour.

II. PROBLEM DESCRIPTION

A. Problem Definition

n jobs must be processed through m stages. Stage k_b ($1 \leq k_b \leq m$) hosts a BPM with capacity C and fixed processing time T ; all other stages host discrete machines. All jobs follow the same processing order across discrete stages (permutation constraint). At the BPM stage, jobs are grouped into batches and processed synchronously, sharing identical start and completion times. The objective is to simultaneously minimize makespan $C_{\max}$ and total carbon emissions TCE.

The problem couples two interdependent decision layers. The *permutation decision* determines the job sequence across discrete stages. The *batching decision* assigns jobs to batches and schedules batch start times. The coupling is bidirectional: the permutation governs job arrival at the BPM, while batching determines downstream completion times; altering either layer can change the optimal choice in the other.

Three special cases are noteworthy. When $n \leq C$, all jobs fit into a single batch and the problem reduces to standard PFSP. When $k_b = 1$, only downstream discrete stages remain; when $k_b = m$, only upstream discrete stages remain. Since PFSP-BPM reduces to standard PFSP when $n \leq C$, it inherits the strong NP-hardness for $m \geq 3$ [3].

B. Notation

Table I defines the core notation. Auxiliary parameters P_k^{proc} , P^{furnace} , α , ε and the MIP linearization variables $W_{i,j,k}$, $V_{i,j,k}$ are introduced alongside their respective formulas.

C. Assumptions

We assume: (1) each stage has exactly one machine; stage k_b hosts a BPM and all other stages host discrete processing machines; (2) all jobs follow the same processing sequence across discrete stages (permutation constraint) and, within a batch, start and complete synchronously; (3) the BPM processing time T is fixed, independent of job type and load size; (4) BPM

TABLE I
CORE NOTATION AND DECISION VARIABLES

Symbol	Definition
n	Number of jobs
m	Number of stages
k_b	Index of the BPM stage ($1 \leq k_b \leq m$)
$p_{i,k}$	Processing time of job i at stage k ($k \neq k_b$)
T	Fixed BPM processing time
C	Maximum BPM capacity (number of jobs)
B	Maximum number of batches ($B \leq n$)
$\pi(j)$	Job at the j -th permutation position
$x_{i,j}$	1 if job i occupies position j
$S_{i,k}$	Start time of job i at stage k (≥ 0)
$C_{i,k}$	Completion time of job i at stage k (≥ 0)
$z_{i,b}$	1 if job i is assigned to batch b
y_b	1 if batch b is active
n_b	Number of jobs in batch b ($n_b = \sum_i z_{i,b}$)
R_b	Start time of batch b (≥ 0)
$C_{\max}$	Makespan (≥ 0)
TCE	Total carbon emissions

capacity is measured in number of jobs with homogeneous job sizes; (5) infinite buffer exists between stages; (6) setup times are negligible and preemption is not permitted; (7) all parameters are deterministic and machines do not fail; (8) all jobs are available at time zero; (9) jobs released to the shop floor have been pre-screened for metallurgical compatibility by the process-planning stage.

D. Schedule Structure Illustration

Fig. 1 shows an optimal Gantt chart for a representative instance with $n = 4$, $m = 3$, $k_b = 2$. The job permutation is $\pi = [J_4, J_1, J_2, J_3]$ with batches $F1=\{J_4\}$, $F2=\{J_1, J_2\}$, $F3=\{J_3\}$. The BPM row (M2) partitions the timeline into batch intervals; downstream row (M3) shows idle periods caused by the permutation constraint and batch synchronization.

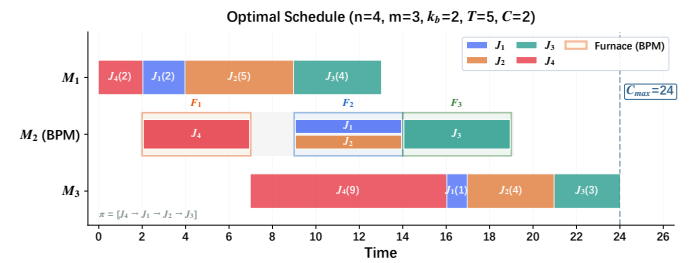


Fig. 1. Optimal PFSP-BPM Gantt chart.

III. MATHEMATICAL MODEL

The MILP model comprises 17 constraint families organized into four groups: permutation and precedence (Group I), batch processing (Group II), objective functions (Group III), and MIP linearization (Group IV). The count excludes the conceptual equation (1).

A. Permutation and Precedence Constraints (Group I)

The permutation constraint requires that, for every position $j \geq 2$ and discrete stage $k \neq k_b$, the job at position j cannot start before the job at position $j - 1$ has finished:

$$S_{\pi(j),k} \geq C_{\pi(j-1),k}, \quad \forall j \in \{2, \dots, n\}, \forall k \neq k_b \quad (1)$$

Constraint (2) is the MIP-linearized form of constraint (1):

$$\sum_{i=1}^n W_{i,j,k} \geq \sum_{i=1}^n V_{i,j-1,k}, \quad \forall j \in \{2, \dots, n\}, \forall k \neq k_b \quad (2)$$

where $W_{i,j,k} = x_{i,j} \cdot S_{i,k}$ and $V_{i,j,k} = x_{i,j} \cdot C_{i,k}$ are auxiliary variables linearized in Section III-D.

Constraint (3) enforces stage precedence (the exclusion $k \neq k_b - 1$ avoids redundancy with batch constraints):

$$S_{i,k+1} \geq C_{i,k}, \quad \forall i, k \neq k_b - 1, k \neq m \quad (3)$$

Constraint (4) defines completion times:

$$C_{i,k} = S_{i,k} + p_{i,k}, \quad \forall i, k \neq k_b \quad (4)$$

Constraint (5) defines the makespan:

$$C_{\max} \geq C_{i,m}, \quad \forall i \quad (5)$$

Constraint (6) enforces the permutation via binary position-assignment variables:

$$\begin{aligned} \sum_{j=1}^n x_{i,j} &= 1, \quad \forall i \\ \sum_{i=1}^n x_{i,j} &= 1, \quad \forall j \end{aligned} \quad (6)$$

B. Batch Processing Constraints (Group II)

Constraint (7) assigns each job to exactly one batch:

$$\sum_{b=1}^B z_{i,b} = 1, \quad \forall i \quad (7)$$

Constraint (8) governs capacity and batch activation:

$$\begin{aligned} \sum_{i=1}^n z_{i,b} &\leq C \cdot y_b, \quad \forall b \\ \sum_{i=1}^n z_{i,b} &\geq y_b, \quad \forall b \end{aligned} \quad (8)$$

Constraint (9) sets the batch start time ($k_b \geq 2$):

$$R_b \geq C_{i,k_b-1} - M_1(1 - z_{i,b}), \quad \forall i, b, k_b \geq 2 \quad (9)$$

When $z_{i,b} = 1$ the inequality is active; when $z_{i,b} = 0$, M_1 renders it vacuous. For $k_b = 1$, $R_b \geq 0$ follows from the variable domain.

Constraints (10) and (11) enforce synchronous batch start ($S_{i,k_b} = R_b$ whenever $z_{i,b} = 1$):

$$S_{i,k_b} \geq R_b - M_2(1 - z_{i,b}), \quad \forall i, b \quad (10)$$

$$S_{i,k_b} \leq R_b + M_2(1 - z_{i,b}), \quad \forall i, b \quad (11)$$

Constraint (12) defines BPM completion:

$$C_{i,k_b} = S_{i,k_b} + T, \quad \forall i \quad (12)$$

Constraint (13) enforces serial batch execution:

$$R_{b+1} \geq R_b + T \cdot y_b, \quad \forall b \quad (13)$$

Constraint (14) explicitly defines batch load, ensuring the linearized TCE has a well-defined variable provenance:

$$n_b = \sum_{i=1}^n z_{i,b}, \quad \forall b \quad (14)$$

Constraint (15) is the symmetry-breaking constraint, eliminating symmetric solutions from batch-index permutations:

$$y_b \geq y_{b+1}, \quad \forall b \in \{1, \dots, B-1\} \quad (15)$$

C. Objective Functions (Group III)

The model minimizes two independent objectives:

$$\min f_1 = C_{\max} \quad (16)$$

$$\min f_2 = \text{TCE} \quad (17)$$

The total carbon emissions comprises two components. The first is a deterministic constant:

$$\text{TCE}_{\text{const}} = \varepsilon \sum_{k \neq k_b} \sum_{i=1}^n p_{i,k} P_k^{\text{proc}} \quad (18)$$

Minimizing TCE is therefore equivalent to minimizing $\text{TCE}_{\text{batch}}$ alone. The second component captures the variable carbon emissions from the BPM:

$$\text{TCE}_{\text{batch}} = \varepsilon \sum_{b=1}^B T \cdot y_b \cdot P^{\text{furnace}} \cdot \left(1 + \alpha \frac{C - n_b}{C}\right) \quad (19)$$

where P^{furnace} is the BPM operating power, $\alpha \geq 0$ is the idle-penalty coefficient reflecting the empty-to-full furnace energy ratio, and $(C - n_b)/C$ is the normalized idle ratio. Because $\text{TCE}_{\text{const}}$ is decision-independent and $\sum_b n_b = n$ is fixed, $\text{TCE}_{\text{batch}}$ depends solely on the active batch count $B_{\text{active}} = \sum_b y_b$:

$$\varepsilon T P^{\text{furnace}} [(1 + \alpha) B_{\text{active}} - \alpha n / C] \quad (20)$$

Thus minimizing TCE amounts to minimizing B_{active} . The bi-objective conflict with makespan arises if and only if the makespan-optimal schedule employs more batches than the carbon-optimal schedule.

D. MIP Linearization

1) *TCE Linearization:* Equation (19) contains the product $y_b \cdot n_b$. Constraint (8) guarantees $y_b \cdot n_b = n_b$: when $y_b = 0$, the second line of constraint (8) forces $n_b = 0$; when $y_b = 1$, $y_b \cdot n_b = n_b$. This yields the linear form:

$$\begin{aligned} \text{TCE} = \varepsilon \sum_{k \neq k_b} \sum_{i=1}^n p_{i,k} P_k^{\text{proc}} \\ + \varepsilon \cdot T \cdot P^{\text{furnace}} \sum_{b=1}^B \left[(1 + \alpha) y_b - \frac{\alpha}{C} n_b \right] \end{aligned} \quad (21)$$

2) *Permutation Constraint Linearization*: Each $W_{i,j,k} = x_{i,j} \cdot S_{i,k}$ is linearized via Big-M ($V_{i,j,k}$ follows the same pattern):

$$\begin{aligned} W_{i,j,k} &\leq S_{i,k} + M_3(1 - x_{i,j}) \\ W_{i,j,k} &\geq S_{i,k} - M_3(1 - x_{i,j}) \\ W_{i,j,k} &\leq M_3 \cdot x_{i,j} \\ W_{i,j,k} &\geq 0 \end{aligned} \quad (22)$$

When $x_{i,j} = 1$, the first two lines of constraint (22) enforce $W_{i,j,k} = S_{i,k}$; when $x_{i,j} = 0$, the last two lines enforce $W_{i,j,k} = 0$. The non-negativity condition (fourth line) prevents inactive terms from assuming negative values that would spuriously tighten constraint (2).

E. Refined Big-M Values

Table II lists the refined Big-M values, derived from problem parameters to tighten the LP relaxation.

TABLE II
REFINED BIG-M VALUES

Constraint	Value
(9) Batch start	$M_1 = \sum_{i'=1}^n \sum_{k'=1}^{k_b-1} p_{i',k'}$
(10)(11) Synchronous start	$M_2 = M_1 + B \cdot T$
(2), (22) MIP linearization	$M_3 = \sum_{i'} \sum_{k'} p_{i',k'} + B \cdot T$

F. Variable Domains and Model Properties

Variable domains: $S_{i,k}, C_{i,k}, R_b, C_{\max} \geq 0$; $z_{i,b}, y_b, x_{i,j} \in \{0, 1\}$; $n_b \in \mathbb{Z}_{\geq 0}$; $W_{i,j,k}, V_{i,j,k} \geq 0$. After linearization, the model is a standard MILP.

IV. COMPUTATIONAL EXPERIMENTS AND VALIDATION

A. Experimental Setup

All experiments ran on an Intel Core i7-12700H CPU with 16GB RAM, Windows 11, using IBM ILOG CPLEX 12.8 under default settings (relative MIP gap tolerance 10^{-4} , single-thread, optimality emphasis). Instance parameters: $p_{i,k} \sim U[1, 10]$, $T \sim U[5, 15]$, $C \sim U\{2, 3\}$, $P_k^{\text{proc}} \sim U[3, 8]$, $P_k^{\text{furnace}} \sim U[15, 25]$, $\alpha \sim U[0.1, 0.3]$, $\varepsilon \sim U[0.01, 0.05]$. These synthetic ranges span plausible carburizing-to-machining time ratios in gear-ring lines. The batch count upper bound is set to $B = \min(n, 2\lceil n/C \rceil)$, tightening the theoretical $B \leq n$ without excluding optimal solutions. Time limits: 300 s (Experiments 1, 3), 3600 s (Experiment 2).

B. Model Correctness Verification (Experiment 1)

Method: Sixteen instances across four groups—Group A ($k_b = 2$, 8 instances), Group B ($k_b = 1$, 3), Group C ($m = 4$, 3), Group D ($k_b = m$, 2)—were solved by both exhaustive enumeration and CPLEX for $\min C_{\max}$ and $\min$ TCE. Validation criterion: $|\Delta C_{\max}| < 10^{-3}$ and $|\Delta \text{TCE}| < 10^{-3}$.

Results: All 16/16 instances passed. Maximum $|\Delta C_{\max}| = 1.14 \times 10^{-13}$ (floating-point level); $|\Delta \text{TCE}| = 0$ (exact match). TCE linearization error $\leq 3.55 \times 10^{-15}$. Table III shows representative results.

TABLE III
MODEL CORRECTNESS VERIFICATION (8 REPRESENTATIVE INSTANCES)

Group	Ins.	n	k_b	T	C	$C_{\max}$ (MILP)	$C_{\max}$ (Enum.)	$\Delta C_{\max}$ ($\times 10^{-14}$)	ΔTCE
A	A1	4	2	5	2	24.0	24.0	4.26	0
A	A4	4	2	15	2	55.0	55.0	0	0
B	B1	4	1	5	2	24.0	24.0	0	0
B	B2	4	1	15	2	48.0	48.0	11.4	0
C	C1	4	2	13	3	41.0	41.0	0	0
C	C3	4	2	15	2	49.0	49.0	0	0
D	D1	4	3	5	2	24.0	24.0	2.13	0
D	D2	4	3	15	2	47.0	47.0	5.68	0

Note: All 16 instances passed. $\Delta C_{\max}$ at floating-point level; $\Delta \text{TCE} = 0$. TCE linearization error $\leq 3.55 \times 10^{-15}$.

These results establish model correctness under all tested parameter combinations, including boundary cases $k_b = 1$ and $k_b = m$.

C. CPLEX Solver Performance Analysis (Experiment 2)

Method: Twelve scale configurations (4×3 to 20×3 , up to 10×5), three random seeds each (36 instances), 3600 s time limit.

Results: Table IV and Fig. 2 summarize the results.

TABLE IV
CPLEX SOLVER PERFORMANCE GRADIENTS (MEAN OF 3 SEEDS PER SCALE)

Scale ($n \times m$)	Vars.	Cons.	Time (s)	Gap (%)	Opt. rate
4×3	133	312	0.1	0.00	3/3
5×3	191	454	0.3	0.00	3/3
6×3	253	604	1.3	0.00	3/3
6×4	349	876	1.8	0.00	3/3
7×3	348	852	32.6	0.00	3/3
8×3	450	1115	452.1	$\leq 0.01^\dagger$	1/3
8×5	738	1929	596.3	0.00	3/3
10×3	691	1746	2414.8	$0.00^\ddagger$	1/3
10×5	1114	2944	> 3600	—	0/3
12×3	953	2409	> 3600	—	0/3
15×3	1426	3604	> 3600	—	0/3
20×3	2535	6526	> 3600	—	0/3

† Gap $\leq 0.01\%$ for two instances (incumbents 44.0, 59.0); optimal for one (65.0). ‡ Optimal for one (83.0); no incumbent or bound for two. — = no incumbent.

Finding 1—A sharp tractability transition at $n = 8$. At $n \leq 7$, all 15 instances reach optimality within one minute. The jump to $n = 8$ is dramatic: mean solution time increases 14-fold and the optimality rate collapses from 100% to 33%. At $n = 8 \times 3$, two instances terminated within 0.01% of the best bound and one reached optimality; at $n = 10 \times 3$, one instance solved to optimality while the other two exhausted the time limit without producing an incumbent. By $n = 12$, no instance yielded an incumbent within one hour. Notably, the $m = 5$ configuration achieves 100% optimality at $n = 8$ despite 64% more variables than $m = 3$, suggesting that

constraint matrix structure, rather than variable count alone, determines effective difficulty.

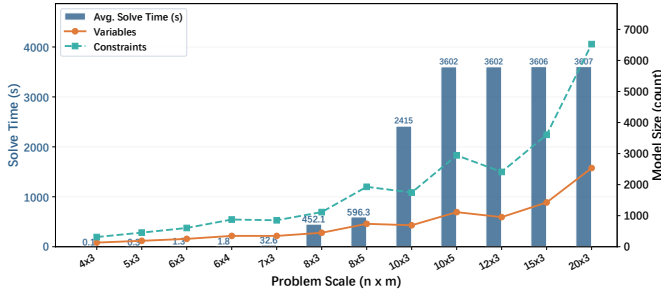


Fig. 2. CPLEX solver performance (dual-axis). Bars: mean solution time; lines: variable and constraint counts.

Model size grows quadratically ($O(n^2)$, $R^2 > 0.99$ for both variables and constraints). Fig. 3 fits the solution time to an exponential model $t = a \cdot e^{b \cdot n}$, yielding $b \approx 1.97$; within the tractable range $n \leq 8$, each additional job increases solution time by approximately a factor of 7.2. Extrapolating from this fit suggests that, under the present MILP formulation with refined Big-M values and default CPLEX settings, the boundary beyond which exact optimality can be certified lies between $n = 8$ and $n = 9$ for $m = 3$.

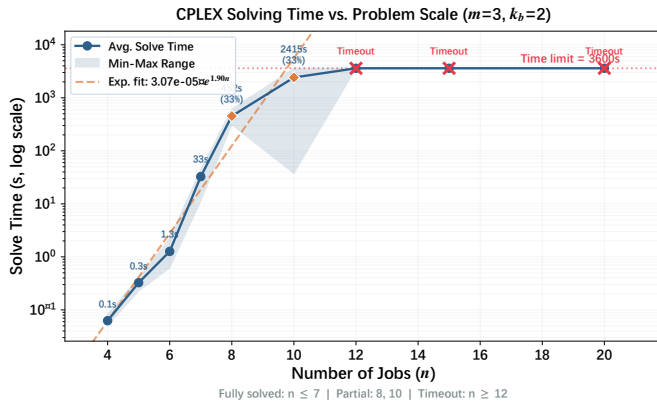


Fig. 3. Exponential solution time growth ($t \propto e^{1.97n}$). Shaded region: range across three seeds.

D. Multi-Objective Pareto Front (Experiment 3)

Method: The ε -constraint method swept the TCE upper bound from $TCE_{\min}$ to $TCE_{\max}$, solving $\min C_{\max}$ at each step and filtering dominated solutions. Both enumeration and the MILP were applied to three instances.

Results: The MILP Pareto fronts match enumeration within 10^{-4} for all three instances (Fig. 4). Instance A ($T = 5$, $C = 2$) produces two non-dominated solutions: (24, 11.13) and (26, 8.52), a 23.5% TCE reduction for two time units of makespan increase. Instances B ($T = 15$) and C ($m = 4$) each produce a single non-dominated point, indicating that multi-objective conflict is parameter-dependent.

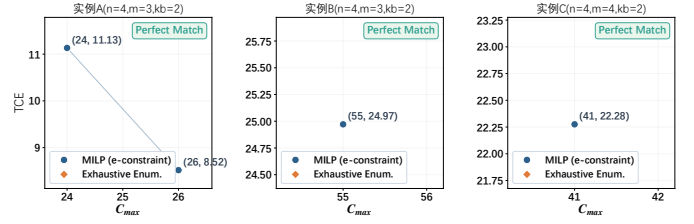


Fig. 4. ε -constraint Pareto fronts. Blue: MILP; orange: enumeration. Instance A exhibits genuine bi-objective trade-off.

E. Comparison with Conventional and Decoupled Baselines (Experiment 4)

Method: We benchmark four strategies against the MILP joint optimum: **NAT** (natural work-order) and **SPT** (shorter jobs first), two conventional sequential-batching rules; **FP-OB** (fixed SPT permutation with optimized batching); and **OP-SB** (optimized permutation with sequential batching). FP-OB and OP-SB are evaluated by exhaustive enumeration on eight representative instances ($n \leq 6$) to isolate the marginal contribution of batching and permutation, respectively.

Results: Table V presents the comparison. On these instances, all strategies share $B_{\text{active}} = \lceil n/C \rceil$, so TCE is invariant; the MILP minimum-makespan solutions coincide with the minimum-batch-count solutions, and the bi-objective trade-off observed in Experiment 3 does not manifest here. Two findings on makespan performance emerge:

Finding 1—Contribution decomposition reveals three patterns. The MILP improves makespan by 4.7%–20.0% over NAT (average 13.6%) and by 0%–14.3% over SPT (average 7.4%). Measuring FP-OB and OP-SB against their common baseline SPT isolates each dimension’s marginal contribution. Three patterns emerge: (i) *permutation-dominant* (e.g., n4m3-kb1, OP-SB reaches MILP with a 14.3% improvement over SPT, while FP-OB yields none); (ii) *batching-dominant* (e.g., n5m3-s42, FP-OB improves by 4.9% via batching alone, while OP-SB yields none); and (iii) *synergistic* (e.g., n4m3-s42, neither FP-OB nor OP-SB alone improves over SPT, yet MILP achieves 7.7%). The remaining two instances (n4m3-s100, n6m3-s42) underscore the parameter-dependent nature of the coupling.

Finding 2—Synergy demonstrates non-separability. In three of eight instances, synergy gains of 4.7%–7.7% show that permutation and batching decisions are non-separable: optimizing either in isolation fails to reach the joint optimum. This justifies the integrated MILP formulation over sequential or decoupled heuristics.

V. CONCLUSION

This paper analyzed the permutation–batching coupling in flow-shop scheduling with an embedded finite-capacity batch processing machine, using gear carburizing as a representative case and treating metallurgical compatibility as a pre-processing condition. A bi-objective MILP model jointly minimizing makespan and total carbon emissions was made

TABLE V
COMPARISON OF CONVENTIONAL RULES, DECOUPLED BASELINES, AND MILP

Instance	$C_{\max}$ (NAT)	$C_{\max}$ (SPT)	$C_{\max}$ (FP-OB)	$C_{\max}$ (OP-SB)	$C_{\max}$ (MILP)	Δ NAT (%)	Δ SPT (%)	Δ FP (%)	Δ OP (%)
n4m3-s42	30	26	26	26	24	20.0	7.7	7.7	7.7
n4m3-s100	56	52	51	51	51	8.9	1.9	0.0	0.0
n4m3-s200	45	45	45	40	40	11.1	11.1	11.1	0.0
n4m3-s300	61	61	59	55	55	9.8	9.8	6.6	0.0
n5m3-s42	46	41	39	41	37	19.6	9.8	4.9	9.8
n6m3-s42	61	49	49	49	49	19.7	0.0	0.0	0.0
n4m4-s42	43	43	43	43	41	4.7	4.7	4.7	4.7
n4m3-kb1	28	28	28	24	24	14.3	14.3	14.3	0.0
Average	—	—	—	—	—	13.6	7.4	6.2	2.8

TCE is identical across all strategies in every instance. Δ NAT, Δ SPT, Δ FP, Δ OP: makespan improvement of MILP over NAT, SPT, FP-OB, and OP-SB (%).

directly solvable by standard solvers through explicit linearization, symmetry breaking, and tightened Big-M values.

Correctness was established through 16-instance enumeration and ε -constraint Pareto-front verification. Three findings emerged. First, exact solution time grows exponentially; optimality becomes difficult to certify beyond $n = 8$, with no incumbent for $n \geq 12$ within one hour, suggesting that industrial-scale problems will require metaheuristics or decomposition algorithms [11], [17]–[20]. Second, TCE depends solely on batching structure; a Pareto trade-off with makespan arises if and only if the makespan-optimal schedule employs more batches than the carbon-optimal schedule (Experiment 3). Third, decoupled-baseline experiments reveal that permutation and batching decisions are non-separable: in three of eight instances, neither dimension alone reaches the MILP optimum, yet joint optimization achieves 4.7%–7.7% additional makespan reduction, confirming that the return on integrated scheduling is structural rather than incidental.

From a managerial perspective, the results yield four actionable insights: (1) permutation and batching are non-separable, and decomposition strategies can forfeit potential makespan improvement; (2) carbon emissions are driven by batching structure, so carbon targets should focus on load consolidation, while makespan targets require joint optimization; (3) accepting additional batches to shorten makespan is justified when deadlines are tight, even at a carbon premium; and (4) joint optimization achieves up to 14.3% makespan reduction over SPT sequencing, and the consistently observed synergy gains indicate that investing in integrated scheduling tools yields structural returns.

This study has several limitations. Exact solutions are reliably obtainable only for $n \leq 7$ under the current formulation; instance parameters are drawn from uniform distributions rather than industry-calibrated data; the model assumes homogeneous jobs with deterministic processing times; and metallurgical compatibility is treated as a pre-processing condition rather than explicit constraints. Future work will proceed along three directions: (1) scalable algorithms, including column-generation or Lagrangian-relaxation decompositions and adap-

tive large-neighbourhood search for industrial-scale instances; (2) heterogeneous and uncertain environments, accounting for gear modules, stochastic processing times, and equipment failures; and (3) metallurgical-compatibility modeling, integrating material composition, case-depth tolerance, and structural/weight similarity into batch formation.

REFERENCES

- [1] S. A. Mansouri, E. Aktas, and U. Besikci, “Green scheduling of a two-machine flowshop: Trade-off between makespan and energy consumption,” *Eur. J. Oper. Res.*, vol. 248, no. 3, pp. 772–788, 2016.
- [2] S. M. Johnson, “Optimal two- and three-stage production schedules with setup times included,” *Naval Res. Logist. Quart.*, vol. 1, no. 1, pp. 61–68, 1954.
- [3] R. Ruiz and C. Maroto, “A comprehensive review and evaluation of permutation flowshop heuristics,” *Eur. J. Oper. Res.*, vol. 165, no. 2, pp. 479–494, 2005.
- [4] G. S. Liu, Y. Zhou, and H. D. Yang, “Minimizing energy consumption and tardiness penalty for fuzzy flow shop scheduling with state-dependent setup time,” *J. Clean. Prod.*, vol. 147, pp. 470–484, 2017.
- [5] D. M. Utama, M. D. Primayesti, S. Z. Umamy, and B. M. N. Kholifa, “Energy-efficient permutation flow shop scheduling problem: A systematic literature review,” *J. Tek. Ind.*, vol. 25, no. 1, pp. 13–54, 2024.
- [6] S. Schulz, M. Schönheit, and J. S. Neufeld, “Multi-objective carbon-efficient scheduling in distributed permutation flow shops under consideration of transportation efforts,” *J. Clean. Prod.*, vol. 365, p. 132551, 2022.
- [7] R. Uzsoy, “Scheduling a single batch processing machine with non-identical job sizes,” *Int. J. Prod. Res.*, vol. 32, no. 7, pp. 1615–1635, 1994.
- [8] C. N. Potts and M. Y. Kovalyov, “Scheduling with batching: A review,” *Eur. J. Oper. Res.*, vol. 120, no. 2, pp. 228–249, 2000.
- [9] Y. Zhang, D. Lei, and J. Wang, “An artificial bee colony with diversified operators for energy-efficient hybrid flow shop scheduling with batch processing machines,” *Eng. Appl. Artif. Intell.*, vol. 136, p. 108930, 2024.
- [10] J. Wang, D. Lei, and H. Tang, “A multi-objective dynamical artificial bee colony for energy-efficient fuzzy hybrid flow shop scheduling with batch processing machines,” *Expert Syst. Appl.*, vol. 259, p. 125244, 2025.
- [11] D. Lei and C. He, “An adaptive shuffled frog-leaping algorithm for flexible flow shop scheduling problem with batch processing machines,” *Appl. Soft Comput.*, vol. 166, p. 112230, 2024.
- [12] T. Gu, S. J. Li, Y. L. Lin, and X. L. Wu, “Research on the re-entrant batch discrete flow shop scheduling for periodic annealing furnace as batch processor,” *J. Mech. Eng.* (in Chinese), vol. 56, no. 2, pp. 220–232, 2020.
- [13] J. W. Fowler and L. Mönch, “A survey of scheduling with parallel batch (p-batch) processing,” *Eur. J. Oper. Res.*, vol. 298, no. 1, pp. 1–24, 2022.
- [14] S. G. Koh, P. H. Koo, J. W. Ha, and J. C. Lee, “Scheduling a batch processing machine with incompatible job families,” *Int. J. Prod. Res.*, vol. 43, no. 17, pp. 3605–3618, 2005.
- [15] G. Dobson and R. S. Nambimadom, “The batch loading and scheduling problem,” *Oper. Res.*, vol. 49, no. 1, pp. 52–65, 2001.
- [16] A. Mencaroni, P. Leyman, B. Raa, S. De Vuyst, and D. Claeys, “Towards net-zero manufacturing: Carbon-aware scheduling for GHG emissions reduction,” *J. Clean. Prod.*, vol. 529, p. 146787, 2025.
- [17] R. Ruiz and T. Stützle, “A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem,” *Eur. J. Oper. Res.*, vol. 177, no. 3, pp. 2033–2049, 2007.
- [18] T. Guo, Y. Mei, M. Zhang, K. Tang, K. Cai, and W. Du, “Enhanced evolution of parallel algorithm portfolio for vehicle routing problem via transfer optimization,” *IEEE Trans. Evol. Comput.*, pp. 1–1, 2025, doi: 10.1109/TEVC.2025.3616385.
- [19] X. Shi, L. L. Minku, and X. Yao, “Evolving memristive reservoir,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 35, no. 10, pp. 13574–13588, 2024, doi: 10.1109/TNNLS.2023.3270224.
- [20] Q.-K. Pan, M. F. Tasgetiren, and Y.-C. Liang, “A discrete differential evolution algorithm for the permutation flowshop scheduling problem,” *Comput. Ind. Eng.*, vol. 55, no. 4, pp. 795–816, 2008.