RESEARCH PAPER

An Integrated Batch Scheduling Model for Sequential Flow Shops in a Production Network Processing a Single Item to Minimize Total Actual Flow Time

Andri Rachmat Kumalasian Nasution[a], Mohammad Mi'radj Isnaini[a], Suprayogi[a], and Abdul Hakim Halim[a,*]

[a]*Faculty of Industrial Technology, Institut Teknologi Bandung, Jl. Ganesha No. 10, Bandung 40132, Indonesia*

**Corresponding author. E-mail: ahakimhalim@itb.ac.id*

ABSTRACT

This study develops an integrated batch-scheduling model for sequential flow shops in a production network processing a single item under a common due date. The model simultaneously determines the number of batches, batch sizes, and a feasible backward schedule to minimize the Total Actual Flow Time while accounting for stage-specific processing and setup times. An independent Gurobi procedure evaluates every integer value for the number of batches in the entire computational domain and solves the corresponding nonconvex model via spatial branch-and-bound with a zero-gap target, subject to solver numerical tolerances. A heuristic is developed to reduce this computational effort by combining a continuous estimate of the number of batches, KKT-based batch sizing, backward-schedule reconstruction, and directional neighborhood search. The computational experiment compares the proposed heuristic with the independently obtained Gurobi reference and a Simulated Annealing benchmark. All 300 benchmark instances are globally resolved by Gurobi over the complete computational domain under the final solver protocol, and the proposed heuristic matches the reported Gurobi objective and selected number of batches at the adopted reporting precision. Simulated Annealing provides an additional independent stochastic comparison. Sensitivity analyses examine demand, processing times, production-network length, and minimum allowable batch size, and a real-data application demonstrates operational relevance under explicitly stated practical limitations.

KEYWORDS: *Production Network; Flow Shop; Batch Scheduling; Heuristic Algorithm; Total Actual Flow Time; Backward Scheduling.*

1. Introduction

This study develops an integrated batch-scheduling model for sequential flow shops in a production network processing a single item to minimize Total Actual Flow Time (TF^a), where all completed parts must be delivered at a common due date positioned at the end of the final processing stage in the production network, from which the schedule is constructed backward. The model simultaneously determines the number of batches, batch sizes, and schedules of the resulting batches at the respective flow shops operated by several companies in the production network. Each flow shop has its own processing and setup times, which may differ across flow shops. These parameters are considered simultaneously because the schedule generated at one flow shop affects batch availability and schedule feasibility at downstream flow shops.

Behnamian and Fatemi Ghomi [1,2], Behnamian [3], Lohmer and Lasch [4], Bagheri Rad and Behnamian [5], Pérez-González and Framiñan [6], and Becker et al. [7] investigated scheduling in production networks through decisions concerning facility assignment, job allocation and sequencing, capacity coordination, blocking, inter-company transfer, and inter-facility schedule control, without treating the number and sizes of production batches as the primary decisions. Recent studies have further examined decentralized flow shops with inter-factory batch delivery, multi-echelon distributed production networks, and distributed hybrid flow-shop batch scheduling [8-10]. Hall and Potts [11] linked scheduling, batching, and delivery decisions within a supply chain. Agnetis et al. [12] further integrated production scheduling and batch delivery by considering fixed departure times and inventory holding costs. Prasetyaningsih et al. [13,14] developed integrated production- and delivery-batch scheduling under a Just-in-Time environment, whereas Rahmawati et al. [15] integrated production and delivery batching for simultaneous multi-job scheduling in a multi-company environment. Rahmawati et al. [16] subsequently developed an integrated formulation linking serial production batching, parallel production batching, and inter-company delivery. These studies demonstrate that operations among interconnected companies are interdependent because the completion of processing at one company determines material availability and processing feasibility at the subsequent company. Differences in processing times, setup times, capacities, batch structures, and delivery policies may generate synchronization difficulties, inter-company waiting, work-in-process accumulation, and excessively early completion of finished products. Consequently, a decision that performs well for one company may not provide the best performance for the production network.

Beyond production scheduling, coordination among production-network members has also been studied through integrated lot-sizing, inventory, and distribution decisions. Karimi and Davoudpour [17] integrated production and delivery scheduling in a multi-factory supply chain while considering stage-dependent inventory holding costs. Goyal [18] developed an integrated inventory model between a single supplier and a single buyer that became one of the foundations of coordinated lot-sizing decisions through the Joint Economic

Lot-Sizing approach. Waller et al. [19] examined Vendor-Managed Inventory as a collaborative arrangement in which the supplier monitors the buyer's inventory and determines replenishment decisions based on shared demand and inventory information. More recently, Salas-Navarro et al. [20] developed a Vendor-Managed Inventory model for a three-layer supply chain involving suppliers, manufacturers, and retailers. Dantzig and Ramser [21] introduced the Vehicle Routing Problem to coordinate vehicle-assignment and delivery-route decisions, whereas Khayya et al. [22] reviewed its development into numerous formulations and solution methods for various logistics conditions. These developments, together with integrated production and delivery-batching studies [11, 13-16], indicate that lot-sizing, inventory, production, and distribution decisions should not be determined independently. However, the network structures considered are generally centered on supplier–buyer, supplier–manufacturer–buyer, or production–distribution relationships. Such structures do not fully represent a multi-company production network in which batches pass sequentially through several flow shops and the completion of a batch at one company determines its availability and processing feasibility at the subsequent company. Integrated scheduling is therefore required across all flow shops in the production network rather than merely coordinating inventory, lot sizes, or distribution between two network members. In this longer structure, the completion of a batch at one company determines material availability, processing start times, and waiting at the subsequent company.

In terms of scheduling orientation and performance measurement, Behnamian and Fatemi Ghomi [1,2], Zhang et al. [23], Pérez-González and Framiñan [6], Becker et al. [7], and recent distributed-production studies [8-10] generally employ completion- or cost-oriented performance measures, including makespan, total completion time, tardiness, resource utilization, energy consumption, transportation cost, and operating cost. Their schedules are commonly constructed from initial job availability times to job completion. Such approaches are appropriate when the scheduling objective is to complete jobs as early as possible, improve resource utilization, reduce tardiness, or control operating and transfer costs. However, in a Just-in-Time environment with a common due date, excessively early completion may prolong the time products remain in the production system, increase work-in-process inventory, and create waiting before delivery. A scheduling approach is therefore required that not only focuses on early completion but also positions production as close as possible to the required delivery time while maintaining schedule feasibility.

Halim and Ohta [24] and Halim et al. [25,26] defined actual flow time as the time measured from the beginning of processing of a part until its due date. Under a common due date, backward scheduling positions each batch as close as possible to the due date without violating schedule feasibility, thereby reducing work-in-process inventory and the waiting time of parts. Halim and Yusriski [27] subsequently developed an integrated batch-scheduling model for a fabrication–assembly system. Yusriski et al. [28] determined the integer number and sizes of batches on a single machine, whereas Yusriski et al. [29,30] incorporated learning, forgetting, and deterioration effects. Zahedi et al. [31,32] integrated batch-production and maintenance scheduling in a two-machine flow shop under a Just-in-Time environment, whereas Yusriski et al. [33] integrated batch scheduling and preventive maintenance on a single machine.

Hidayat et al. [34] developed a batch-scheduling model for multiple heterogeneous batch processors, whereas Hidayat et al. [35] considered a single batch processor with additional setup times for multiple items. Halim et al. [36] developed a flow-shop model with multiple batch-processing machines. Suryadhini et al. [37,38] developed three-stage flow-shop models involving job processors, batch processors, sampling inspection, and heterogeneous machines. Maulidya et al. [39,40] developed hybrid flow-shop models involving assembly structures, unique and common components, and multiple due dates. Yusriski et al. [41] considered multiple jobs and multiple due dates; Yusriski et al. [42] investigated a two-machine flow shop, and Kurniawan et al. [43] incorporated preprocessing and time-changing effects. Sibarani et al. [44,45] developed models for multi-item flow shops, whereas Yusriski et al. [46] addressed unrelated parallel machines with resource constraints and sequence-dependent setup times, and Yusriski et al. [47] developed a heuristic for a flexible flow shop. These studies demonstrate that TF^a and backward scheduling have been developed for various machine and shop-floor configurations. However, the available formulations remain primarily focused on a single shop floor or a single production-system configuration and therefore do not represent batch flows through several flow shops arranged sequentially within a production network [48-52].

Research on production networks and batch scheduling reveals a specific gap. Production-network studies represent operational interdependencies, capacity coordination, production and delivery batching, inventory, and inter-company transfers, but generally use completion-oriented scheduling and performance measures such as makespan, completion time, tardiness, and cost. In contrast, batch-scheduling studies that minimize TF^a provide methods for determining the number of batches, batch sizes, and backward schedules, but are still predominantly applied to a single shop floor or a single production-system configuration. Limited attention has therefore been given to an integrated model that determines the number of batches, continuous batch sizes, and batch schedules for several sequential flow shops in a production network, while simultaneously considering the different processing and setup times of all stages and applying backward scheduling to minimize TF^a under a common due date.

To address this gap, the present study develops an integrated batch-scheduling model for sequential flow shops in a production network. The model simultaneously determines the number of batches, continuous batch sizes, and a feasible backward schedule that minimizes Total Actual Flow Time under a common due date positioned at the end of the final processing stage. The actual setup times, processing times, demand, common due date, and minimum allowable batch size are used to establish the complete computational range for the number of batches. An equal-sized reference partition is introduced only to obtain an analytical characterization that guides the heuristic search; it is not imposed on the final continuous batch-size decisions. For the independent solver-based reference, every integer value for the number of batches in the complete computational domain is evaluated using spatial branch-and-bound through Gurobi Optimizer under a

zero-gap target, subject to solver numerical tolerances. The proposed heuristic reduces this computational effort by combining a continuous estimate of the number of batches, Karush-Kuhn-Tucker conditions for batch sizing, backward-schedule reconstruction, and a directional neighborhood search over adjacent integer values for the number of batches.

2. Mathematical Model

This section presents the problem description and assumptions, notation, TF^a formulation, determination of the maximum number of batches, and the mathematical formulation of the proposed model. The development first establishes the relationship among the number of batches, batch sizes, processing times, setup times, and the available horizon up to the common due date. A common upper bound on the number of batches is then derived from the minimum batch-size requirement and the scheduling-horizon condition of each processing stage within the sequential flow-shop network. The resulting bound defines the computational search range used by both the Gurobi-based and heuristic solution procedures.

2.1. Problem Description and Assumptions

A production network dealt with in this study consists of multiple interconnected flow shops. Each flow shop consists of one or more shop floors arranged according to the required processing sequence. Each shop floor performs one processing activity and is represented as one processing stage in the mathematical model. Therefore, the terms shop floor and processing stage refer to the same indexed processing unit k. Let c denote the company index and let $c(k)$ denote the company operating processing Stage k. Multiple consecutive processing stages may therefore be operated by the same company. The total demand n is divided into N batches, where Q_i denotes the continuous size of Batch i. The output of Stage k becomes the input to Stage $k+1$. Each of the n units of a single item must pass through all processing stages and be completed at a common due date d positioned at the end of the final processing stage in the production network, from which the schedule is constructed backward. Under backward indexing, at the shop floor k, Batch 1 denotes the first-positioned batch, whereas Batch N denotes the batch processed earliest in forward time. The actual processing time (t_k) and setup time (s_k) of every stage are considered simultaneously in determining the number of batches, batch sizes, and the resulting backward schedule.

The schedule is constructed backward from the common due date. Processing is non-preemptive: once processing of a batch begins on the shop floor, it continues without interruption until completion of all parts in the batch. Batch splitting, batch merging, batch overtaking, and lot streaming are not permitted. Setup is anticipatory, meaning that setup for a batch may be completed before the corresponding batch arrives, provided that the immediately preceding batch has completed processing. The setup for the first batch processed is assumed to have been completed before the scheduling horizon. Transportation time between successive shop floors is assumed negligible, and enough temporary storage is available between shop floors.

For a specified number of batches N and $Q_i, i = (Q_1, \dots, Q_N)$, the processing starts, and completion times are generated deterministically through the backward-scheduling recursion. The objective is to determine a common number of batches N, a common Q_i maintained across all stages, and the corresponding backward schedule that minimizes TF^a, while satisfying demand balance, minimum batch-size requirements, inter-stage precedence, same-shop-floor sequencing, and schedule-feasibility constraints.

2.2. Notation

The notation used throughout the proposed model is summarized in Table 1. The symbols are grouped into indices, parameters, decision variables, schedule and waiting quantities, analytical quantities, and the objective function. The notation N_k and $Q_{i,k}$ is introduced only as a generalized representation used to establish Proposition 1; the proposed model subsequently uses N and Q_i across all stages.

Tab. 1. Notation used in the proposed model

Symbol	Definition
Index	
k	Shop-floor (processing-stage) index, $k = 1, \dots, K$
i	Batch index under backward scheduling
c	Company Index, $c(k)$: company operating Shop Floor k
Parameter	
K	Total number of shop floors (processing stages) in the production network
n	Total demand
d	Common due date
t_k	Unit processing time at Shop Floor k
s_k	Sequence-independent setup time at Shop Floor k
ε	Minimum allowable batch size
Generalized batch representation	
N_k	Number of batches represented at Shop Floor k in the generalized production-network structure
$Q_{i,k}$	Continuous size of Batch i represented at Shop Floor k, $i = 1, \dots, N_k$
Decision variable	
N	The number of batches; a positive integer
Q_i	Continuous size of Batch i, $Q_i \geq \varepsilon$, $i = 1, \dots, N$
Schedule and derived quantity	
$B_{k,i}$	Processing start time of Batch i at Shop Floor k
$C_{k,i}$	Processing completion time of Batch i at Shop Floor k
F_i^a	Actual flow time of Batch i, measured from $B_{1,i}$ to the common due date
$w_{k,i}$	Physical waiting time of Batch i between Shop Floors k and $k+1$
W_k	Total batch-size-weighted waiting at the boundary between Shop Floors k and $k+1$
TWW	Total batch-size-weighted waiting between successive shop floors
PTF_k	Processing-time contribution of Shop Floor k to Total Actual Flow Time
FGW	Batch-size-weighted finished-goods waiting after the final shop floor until the common due date
RBW	Re-batching Waiting caused by batch splitting, merging, or re-partitioning; $RBW = 0$ under the proposed production structure
Reference quantity	
P	Total unit processing time across the production network, $P = \sum_{r=1}^{K} t_r$
s_c	Reference setup time, selected as the largest setup time among all shop floors
t_c	Reference processing time associated with the shop floor having s_c
Analytical quantity for the number of batches	
$H(N)$	Equal-batch horizon under shop-floor parameters
$H_k(N)$	Equal-batch horizon associated with Shop Floor k
a, b, c	Coefficients of the quadratic feasibility condition
a_k, b_k, c_k	Coefficients of the shop-floor-specific quadratic feasibility condition
D	Discriminants of shop-floor-specific quadratic equations
D_k	Discriminant of the quadratic feasibility condition associated with Shop Floor k

Symbol	Definition
N^L, N^U	Continuous lower and upper roots for the reference structure
N^{LB}, N^{UB}	Computational integer lower and upper bounds for the complete search domain
N_k^L, N_k^U	Continuous lower and upper roots associated with Shop Floor k
$N_{\min}^F, N_{\max}^F$	Reference integer lower and upper bounds for the identical-stage reference case
$N_{\min}^A, N_{\max}^A$	Analytical integer lower and upper bounds obtained from the intersection of the shop-floor-specific intervals
Objective function	
$TF^a(N, \mathbf{Q})$	Total Actual Flow Time minimized by the proposed model

2.3. Total Actual Flow Time

Following the actual-flow-time concept introduced by Halim and Ohta [24], the actual flow time of a part is measured from the beginning of its processing until its common due date. In a general production-network structure, TF^a may consist of the processing-time contribution, waiting caused by changes in batch formation between successive shop floors, waiting between successive shop floors after a batch has become available, and finished-goods waiting until the common due date. Accordingly, its general decomposition is expressed as

$$TF^a = PTF + RBW + TWW + FGW. \quad (1)$$

where PTF denotes the batch-size-weighted processing-time contribution across all shop floors, RBW denotes Re-batching Waiting caused by batch splitting, merging, or re-partitioning between successive shop floors, TWW denotes the total batch-size-weighted waiting between successive shop floors, and FGW denotes the batch-size-weighted finished-goods waiting after processing at the final shop floor until the common due date.

Proposition 1. Under the production structure considered in this study, batch splitting, merging, re-partitioning, and lot streaming are not permitted. Each batch is transferred intact from one shop floor to the next while retaining its identity and size. Consequently, a one-to-one batch correspondence is maintained throughout the production network, such that

$$N_k = N, \qquad k = 1, \dots, K, \quad (2)$$

and

$$Q_{i,k} = Q_i, \qquad i = 1, \dots, N, \quad k = 1, \dots, K. \quad (3)$$

Because no change in batch formation occurs between successive shop floors,

$$RBW = 0. \quad (4)$$

The result follows from material conservation and intact batch transfer. Every batch completed at Shop Floor k corresponds to exactly one batch entering Shop Floor $k+1$, and the corresponding batch contains the same units and retains the same size. Applying this relationship successively across all shop-floor boundaries produces Equations (2) and (3). Because no unit waits for splitting, merging, or re-partitioning into a different batch, Re-batching Waiting is zero. The complete proof of Proposition 1 is provided in Appendix A.

Proposition 1 describes the production structure considered in this study. It does not constitute an optimality claim for a broader production-network problem in which re-batching is permitted.

Based on Proposition 1, the general decomposition in Equation (1) reduces to

$$TF^a = PTF + TWW + FGW. \quad (5)$$

Therefore, one common number of batches N and common batch sizes Q_i are used throughout the production network.

In the production network considered in this study, the actual flow time of Batch i is measured from the beginning of its processing at Shop Floor 1 until the common due date:

$$F_i^a = d - B_{1,i}. \quad (6)$$

Because Batch i contains Q_i units, its contribution to TF^a is

$$TF_i^a = (d - B_{1,i})Q_i. \quad (7)$$

The TF^a of all batches is therefore

$$TF^a(N, \mathbf{Q}) = \sum_{i=1}^{N} (d - B_{1,i})\, Q_i. \quad (8)$$

This measure represents the cumulative actual flow time of all units from the beginning of processing at the first shop floor until delivery at the common due date. The processing-time contribution of Shop Floor k is

$$PTF_k = t_k \sum_{i=1}^{N} Q_i^2. \quad (9)$$

The physical waiting time of Batch i between Shop Floors k and $k+1$ is

$$w_{k,i} = B_{k+1,i} - C_{k,i}, \qquad k = 1, \dots, K-1. \quad (10)$$

The total batch-size-weighted waiting at the boundary k is

$$W_k = \sum_{i=1}^{N} w_{k,i}\, Q_i. \quad (11)$$

The total batch-size-weighted waiting between successive shop floors is

$$TWW = \sum_{k=1}^{K-1} W_k. \quad (12)$$

Finished goods waiting represents the time during which completed units wait after processing at the final shop floor until the common due date:

$$FGW = \sum_{i=1}^{N} (d - C_{K,i})\, Q_i. \quad (13)$$

Accordingly, TF^a can be decomposed as

$$TF^a = \sum_{k=1}^{K} P\, TF_k + TWW + FGW. \quad (14)$$

Equation (14) is equivalent to the direct expression in Equation (8). When the schedules of all shop floors are completely synchronized, waiting between successive shop floors is zero. When differences in processing and setup parameters prevent complete synchronization, positive waiting may arise. Such waiting remains a component of TF^a and is not treated as a separate allocation objective in the present model.

2.4. Mathematical Formulation

Based on Proposition 1, one common number of batches N and common batch sizes Q_i are maintained throughout the production network. Because $RBW = 0$, the mathematical model minimizes TF^a using the processing-time, inter-shop-floor waiting, and finished-goods waiting components established in Section 2.3. The mathematical model is formulated as follows:

$$\min\ TF^a(N, \mathbf{Q}) = \sum_{k=1}^{K} P\, TF_k + TWW + FGW \quad (15)$$

subject to:

$$\sum_{i=1}^{N} Q_i = n, \quad (16)$$

$$Q_i \geq \varepsilon, \qquad i = 1, \dots, N, \tag{17}$$
$$N \in \{1,2,3, \dots\} \tag{18}$$
$$N^{LB} \leq N \leq N^{UB}, \tag{19}$$
$$C_{K,1} = d, \tag{20}$$
$$B_{K,1} = C_{K,1} - t_K Q_1, \tag{21}$$
$$C_{K,i} = B_{K,i-1} - s_K, \qquad i = 2, \dots, N, \tag{22}$$
$$B_{K,i} = C_{K,i} - t_K Q_i, \qquad i = 2, \dots, N, \tag{23}$$
$$C_{k,1} = B_{k+1,1}, \qquad k = 1, \dots, K-1, \tag{24}$$
$$B_{k,1} = C_{k,1} - t_k Q_1, \qquad k = 1, \dots, K-1, \tag{25}$$
$$C_{k,i} = \min\{B_{k+1,i}, B_{k,i-1} - s_k\}, \quad k = 1, \dots, K-1, \quad i = 2, \dots, N, \tag{26}$$
$$B_{k,i} = C_{k,i} - t_k Q_i, \quad k = 1, \dots, K-1, \quad i = 2, \dots, N, \tag{27}$$
$$B_{k,i} \geq 0, \qquad k = 1, \dots, K, \quad i = 1, \dots, N. \tag{28}$$

Equation (15) minimizes TF^a, using the processing-time, inter-shop-floor waiting, and finished-goods waiting components established in Section 2.3. Re-batching Waiting does not appear in the model-specific objective because Proposition 1 establishes that $RBW = 0$ under the production structure considered in this study.

Equation (16) ensures that the sum of all batch sizes equals the total demand. Equation (17) requires every batch to satisfy the minimum allowable batch size. Equation (18) specifies N, is a positive integer, whereas Equation (19) restricts N to the established computational domain. The upper computational bound used in Equation (19) is derived from necessary feasibility conditions associated with the minimum allowable batch size and the available scheduling horizon, as presented in Section 2.5.1.

Equations (20)–(23) construct the backward schedule at the final shop floor. Equation (20) positions Batch 1 such that its completion coincides with the common due date. Equation (21) determines the corresponding processing start time. Equations (22) and (23) then position the remaining batches successively before Batch 1 by considering the setup and processing times at the final shop floor.

Equations (24)–(27) reconstruct the backward schedules at the preceding shop floors. Equations (24) and (25) determine the completion and processing start times of Batch 1, respectively. Equation (26) determines the completion time of each subsequent batch by comparing downstream precedence and same-shop-floor sequencing and selecting the earlier feasible completion time. Equation (27) determines the corresponding processing start time.

Equation (28) requires all processing start times to be nonnegative. Together with the same-shop-floor sequencing and inter-shop-floor precedence relationships, this condition ensures that the reconstructed backward schedule can be accommodated within the available scheduling horizon.

Because N determines the number of batch-size and scheduling variables, only a finite and valid range of integer values needs to be considered when solving the model. The following section therefore characterizes the search domain for N. It first derives the computational domain from necessary feasibility conditions and subsequently introduces an equal-sized reference partition to obtain an analytical characterization that is used to guide the heuristic search.

2.5. Solution-Search Framework

Because the number of batches $\boldsymbol{N}$ determines the dimensions of the batch-size and scheduling variables, a finite integer search domain is first established from necessary feasibility conditions. Within this domain, an equal-sized reference partition is used to derive analytical bounds that guide the heuristic search. These analytical bounds are not imposed on the final unequal continuous batch-size solution.

2.5.1. Computational Domain for N

The mathematical formulation requires N to be a positive integer. However, demand balance, the minimum allowable batch size, and the available scheduling horizon impose necessary restrictions on the values of N that can produce a feasible solution. These restrictions are used to establish the computational domain evaluated by the solution procedures.

The lower computational bound is $N_{LB} = 1$. Because each batch must satisfy $Q_i \geq \varepsilon$ and the demand-balance constraint requires $\sum Q_i = n$, the number of batches must satisfy $N\varepsilon \leq n$. Accordingly, the upper bound imposed by the minimum allowable batch size is $N_\varepsilon = \lfloor n/\varepsilon \rfloor$.

A further restriction is obtained from the available scheduling horizon at each Shop Floor k. Define the cumulative unit processing times upstream and downstream of Shop Floor k, respectively, as

$$T_k^U = \sum_{r=1}^{k-1} t_r \tag{29}$$

and

$$T_k^D = \sum_{r=k+1}^{K} t_r \tag{30}$$

where $T_1^U = 0$ and $T_K^D = 0$.

For a schedule containing N batches, all n units must be processed at the Shop Floor k, requiring a total processing time of $t_k n$. In addition, $N-1$ setup activities must be accommodated within the scheduling horizon because the setup for the first processed batch is assumed to have been completed before the scheduling horizon. The batches positioned at the two ends of the backward schedule must also accommodate the processing requirements associated with the upstream and downstream shop floors. Therefore, a necessary condition for schedule feasibility at Shop Floor k is

$$Q_N T_k^U + t_k n + (N-1)s_k + Q_1 T_k^D \leq d \tag{31}$$

Because the minimum allowable batch-size requirement gives $Q_1 \geq \varepsilon$ and $Q_N \geq \varepsilon$, a necessary condition that is independent of the unknown batch-size distribution is

$$t_k n + (N-1)s_k + \varepsilon(T_k^U + T_k^D) \leq d \tag{32}$$

For $s_k > 0$, rearranging Equation (32) with respect to N gives

$$N \leq 1 + \frac{d - t_k n - \varepsilon(T_k^U + T_k^D)}{s_k} \tag{33}$$

Accordingly, the shop-floor-specific computational upper bound is

$$\overline{N_k} = \left\lfloor 1 + \frac{d - t_k n - \varepsilon(T_k^U + T_k^D)}{s_k} \right\rfloor \tag{34}$$

When $s_k = 0$, the corresponding shop floor does not generate a finite upper bound through the setup term. In this case, the necessary scheduling-horizon condition reduces to

$$t_k n + \varepsilon(T_k^U + T_k^D) \leq d \tag{35}$$

If Equation (35) is violated, the instance has no feasible schedule. Otherwise, the corresponding shop floor does not further restrict the number of batches.

Considering both the minimum batch-size restriction and the scheduling-horizon conditions of all shop floors, the common computational upper bound is

$$N_{UB} = \min\left\{N_{\varepsilon}, \min_{k:s_k>0} \overline{N_k}\right\} \tag{36}$$

Hence, the computational domain for N is

$$1 \le N \le N_{UB} \tag{37}$$

The computational upper bound N_{UB} is a necessary feasibility bound. Any integer value $N > N_{UB}$ violates at least one of the minimum-batch-size or scheduling-horizon conditions and therefore cannot produce a feasible solution. Conversely, $N \le N_{UB}$ does not by itself guarantee feasibility because the corresponding Q_i and complete backward schedule must still satisfy all constraints of the mathematical model.

The computational domain in Equation (37) is the complete outer search domain used by both solution approaches. The Gurobi-based procedure evaluates the integer values of N within this domain, whereas the heuristic uses additional analytical information to identify a more focused initial search region.

2.5.2. Equal-Sized Reference Partition

Although the computational domain provides a finite range for N, evaluating every integer value within this domain may require substantial computational effort, particularly as N_{UB} increases. An analytical characterization is therefore developed to provide additional guidance for the heuristic search.

In the complete mathematical model, the batch sizes $Q_1, Q_2, \dots, Q_N$ are continuous decision variables and are allowed to differ. If these batch sizes are retained as independent variables when analytically characterizing the number of batches, the scheduling-horizon relationships depend simultaneously on N and all Q_i. Moreover, the number of batch-size variables Q_i changes with N. Consequently, obtaining a general closed-form expression solely in terms of N becomes difficult.

To obtain a tractable analytical representation, an equal-sized reference partition is introduced for a specified positive value of N:

$$Q_i = q_N = \frac{n}{N}, \qquad i = 1, \dots, N. \tag{38}$$

This equal-sized partition is used only as an analytical reference. It converts the batch-size terms in the scheduling-horizon relationships into functions of N, thereby allowing the number-of-batches range to be characterized analytically. Equation (38) does not impose equal batch sizes on the final optimization model. Once a value of N is evaluated by the solution procedure, the equal-size reference condition is released, and the batch sizes are reoptimized as continuous decision variables subject to the complete mathematical formulation.

2.5.3. Analytical Bounds for the Number of Batches

Because every unit passes successively through all K shop floors, the total unit processing time across the production network is defined as

$$P = \sum_{k=1}^{K} t_k \tag{39}$$

The parameter P represents the cumulative unit processing-time contribution of all shop floors. Based on Equation (9), the total processing-time contribution to Total Actual Flow Time can therefore be expressed as

$$\sum_{k=1}^{K} PTF_k = \sum_{k=1}^{K} t_k \sum_{i=1}^{N} Q_i^2 = P \sum_{i=1}^{N} Q_i^2. \tag{40}$$

Thus, P follows directly from the additive processing-time contribution of the production network and is not obtained by averaging the unit processing times or by selecting the processing time of only one shop floor.

To illustrate the analytical structure, first consider a reference condition in which all shop floors have the same unit processing time and setup time:

$$t_k = t, \qquad s_k = s, \qquad k = 1, \dots, K. \tag{41}$$

Under this reference condition, Equation (39) becomes

$$P = Kt. \tag{42}$$

For a specified N, substituting the equal-sized reference partition in Equation (38) gives the required scheduling horizon

$$H(N) = tn + (N-1)s + \frac{(K-1)tn}{N}. \tag{43}$$

The equal-sized reference schedule satisfies the available horizon when

$$H(N) \le d. \tag{44}$$

Because $N > 0$, multiplying Equation (44) by N and collecting the terms gives

$$sN^2 + (tn - s - d)N + (K-1)tn \le 0. \tag{45}$$

Define

$$a = s, \qquad b = tn - s - d, \qquad c = (K-1)tn. \tag{46}$$

For $s > 0$, the discriminant is

$$D = b^2 - 4ac = (d + s - tn)^2 - 4s(K-1)tn. \tag{47}$$

When $D \ge 0$, the continuous lower and upper roots are

$$N^L = \frac{d + s - tn - \sqrt{D}}{2s}, \qquad N^U = \frac{d + s - tn + \sqrt{D}}{2s}. \tag{48}$$

Because the coefficient of N^2 is positive for $s > 0$, the quadratic function is convex, and the horizon condition in Equation (45) is satisfied between the two roots. The corresponding integer lower bound is

$$N_{\min}^F = \max\{1, \lceil N^L \rceil\}, \tag{49}$$

and the corresponding integer upper bound is

$$N_{\max}^F = \min\left\{\lfloor N^U \rfloor, \left\lfloor \frac{n}{\varepsilon} \right\rfloor\right\} \tag{50}$$

The reference interval is nonempty when

$$N_{\min}^F \le N_{\max}^F. \tag{51}$$

If $s = 0$, the corresponding horizon condition becomes linear and is solved directly. The preceding reference case illustrates the opposing effects associated with increasing the number of batches. Increasing N decreases the equal-sized reference batch n/N, thereby reducing the processing time required for one batch to pass through the remaining shop floors. At the same time, increasing N increases the number of setup activities that must be accommodated within the available scheduling horizon. These opposing effects generate the quadratic relationship with respect to N.

The derivation is then extended to the general production-network structure by retaining the actual unit processing time t_k and setup time s_k of every Shop Floor k. For Shop Floor k, the equal-sized reference horizon is

$$H_k(N) = t_k n + (N-1)s_k + \frac{(P - t_k)n}{N}. \tag{52}$$

The first term represents the time required by Shop Floor k to process the complete demand. The second term represents the $N-1$ setup activities occurring within the scheduling horizon at Shop Floor k. The third term represents the processing time of one equal-sized reference batch across the remaining shop floors.

The shop-floor-specific horizon condition is

$$H_k(N) \le d. \tag{53}$$

Because $N > 0$, multiplying Equation (53) by N and collecting the terms gives

$$s_k N^2 + (t_k n - s_k - d)N + (P - t_k)n \le 0. \tag{54}$$

Define the shop-floor-specific coefficients as

$$a_k = s_k, \qquad b_k = t_k n - s_k - d, \qquad c_k = (P - t_k)n. \tag{55}$$

For $s_k > 0$, the discriminant is

$$D_k = b_k^2 - 4a_k c_k = (d + s_k - t_k n)^2 - 4s_k(P - t_k)n. \tag{56}$$

A positive discriminant produces two real roots, a zero discriminant produces one repeated root, and a negative discriminant indicates that the equal-sized quadratic horizon condition has no real interval for Shop Floor k.

When $D_k \ge 0$, the continuous lower root is

$$N_k^L = \frac{d + s_k - t_k n - \sqrt{D_k}}{2s_k}, \tag{57}$$

and the continuous upper root is

$$N_k^U = \frac{d + s_k - t_k n + \sqrt{D_k}}{2s_k}. \tag{58}$$

The common analytical integer lower bound is obtained from the largest shop-floor-specific lower root:

$$N_{\min}^A = \max\left\{1, \left\lceil \max_{k=1,\dots,K} N_k^L \right\rceil\right\}. \tag{59}$$

The analytical integer upper bound is obtained from the smallest shop-floor-specific upper root together with the minimum-batch-size restriction:

$$N_{\max}^A = \min\left\{\left\lfloor \min_{k=1,\dots,K} N_k^U \right\rfloor, \left\lfloor \frac{n}{\varepsilon} \right\rfloor\right\}. \tag{60}$$

The analytical integer bounds $N_{\min}^A$ and $N_{\max}^A$, derived under the equal-Q_i reference in Equation (38), define a nonempty interval when

$$N_{\min}^A \le N_{\max}^A. \tag{61}$$

If $s_k = 0$, the corresponding shop-floor-specific horizon condition becomes linear and is solved directly.

The quadratic form in Equation (54) results from the two opposing consequences of increasing N. The equal-sized reference batch decreases as n/N, thereby reducing the processing time required for one batch to pass through the remaining shop floors. At the same time, the number of setup activities increases as $N-1$. After the positive denominator N is eliminated, these opposing effects produce a convex quadratic condition for each shop floor with $s_k > 0$.

When all processing stages have identical processing and setup parameters, the stage-specific horizon conditions coincide. In the general production-network case, the intersection of the stage-specific conditions defines the common analytical integer bounds $N_{\min}^A$ and $N_{\max}^A$ for N under the equal-Q_i reference in Equation (38). When processing and/or setup parameters differ among stages, each stage generates its own quadratic horizon condition, and their intersection determines these analytical bounds for the production network.

Therefore, when $N_{\min}^A \le N_{\max}^A$, the analytical interval is nonempty and identifies integer values of N for which the equal-Q_i reference satisfies the scheduling-horizon condition at every processing stage. These analytical bounds must be distinguished from the complete computational domain in Equation (37). The computational domain is derived from necessary feasibility conditions and defines the outer range considered by the solution procedures, whereas $N_{\min}^A$ and $N_{\max}^A$ are derived only from the equal-Q_i reference and are used to guide the heuristic search. Feasible unequal-Q_i solutions may therefore exist outside the analytical interval provided that they remain within the computational domain and satisfy the complete mathematical formulation.

Based on this framework, the Gurobi-based procedure evaluates every integer value of N in the complete computational domain without using $N_{\min}^A$ or $N_{\max}^A$ to exclude candidates. The proposed heuristic uses these analytical bounds only to position its initial search and subsequently determines the actual continuous Q_i values without imposing the equal-Q_i reference in Equation (38). The Gurobi-based and heuristic solution procedures are presented in Sections 3 and 4, respectively. Further analytical derivations supporting the number-of-batches characterization are provided in Appendix B.

3. Solution Procedure Using Spatial BnB

The mathematical model in Section 2 is solved using spatial branch-and-bound implemented in Gurobi Optimizer. Because the number of batches N changes the dimensions of the model, a separate nonconvex optimization model is solved for each integer value of N within the computational domain established in Section 2.5. For each N, Gurobi determines Q_i, $B_{k,i}$, and $C_{k,i}$ while minimizing Total Actual Flow Time, TF^a. This section presents the spatial BnB characteristics, Gurobi implementation and computational protocol, enumeration procedure for N, numerical illustration, and computational evaluation.

3.1. Characteristics of the Spatial BnB Procedure

For a given number of batches N, the model contains continuous batch-size and scheduling variables. Products between batch sizes and scheduling variables generate quadratic and bilinear relationships; consequently, the optimization problem for each N is a nonconvex quadratic optimization problem.

Spatial branch-and-bound addresses this nonconvex structure by constructing globally valid relaxations over bounded variable domains. The relaxations provide lower bounds on the objective value, whereas feasible incumbent solutions provide upper bounds for the minimization problem. A node is eliminated when its relaxation is infeasible or cannot improve the incumbent within the prescribed tolerance. Otherwise, selected variable domains are partitioned and tighter relaxations are generated. The procedure continues until the prescribed optimality criterion is satisfied or another termination condition is reached [53,54].

3.2. Implementation Using Gurobi Optimizer and Computational Protocol

The mathematical model was implemented in Python through the Gurobi Python interface and executed in Jupyter Notebook.

Gurobi Optimizer 13.0.1 was run on a 64-bit Windows 10 computer equipped with a 12th Gen Intel Core i7-12700H processor, 14 physical cores, 20 logical processors, and 32 GB of RAM. The implementation used 20 solver threads and $Seed = 0$ to support reproducibility.

The final solver protocol used $NonConvex = 2, MIPGap = 0$, $MIPGapAbs = 0$, $FeasibilityTol = 10^{-9}$, $OptimalityTol = 10^{-9}, NumericFocus = 3, Threads = 20$, and $Seed = 0$, with no user-imposed finite time limit. For a minimization problem, Gurobi maintains a feasible incumbent as an upper bound and a globally valid best bound as a lower bound while spatial branch-and-bound proceeds. Because the computation is performed in floating-point arithmetic, the zero-gap target is interpreted subject to Gurobi's numerical tolerances [55].

If the model corresponding to a candidate number of batches initially returned INF_OR_UNBD, the same mathematical model was re-solved after setting $DualReductions = 0$. This retry changes the presolve strategy used to distinguish infeasibility from unboundedness but does not change the mathematical formulation. The final status, objective value, global bound, runtime, node information, Q_i, and feasibility-audit results were recorded for every evaluated integer number of batches.

A benchmark instance is classified as globally resolved over the complete computational domain only when every integer value for the number of batches in that domain has been attempted, and the model corresponding to every candidate ends with either OPTIMAL or INFEASIBLE status. Any unresolved candidate prevents a complete-domain global-resolution claim for that instance. Objective values and Q_i shown in reporting tables are rounded for presentation, whereas solver-side feasibility and numerical validation use the available full-precision computational outputs.

3.3. Enumeration Procedure for the Number of Batches

Because N changes the dimensions of the mathematical model, a separate nonconvex optimization model is solved for each integer value within the computational domain:

$$N = 1,2, \dots, N_{UB} \tag{62}$$

For each value of N, Gurobi determines Q_i, $B_{k,i}$, and $C_{k,i}$ while minimizing TF^a subject to the mathematical formulation in Section 2.4. The computational protocol defined in Section 3.2 is applied to the complete enumeration. After all retained values of N have been evaluated, the solution with the smallest objective value is selected:

$$N^G = \text{argmin}_N TF_N^a \tag{63}$$

The complete procedure is summarized in Algorithm 1.

Algorithm 1. Gurobi-Based Solution Procedure

Step 0. Initialize. Read K, n, d, ε, t_k, and s_k, $k = 1, \dots, K$. Set the Gurobi parameters: NonConvex = 2, MIPGap = 0, MIPGapAbs = 0, FeasibilityTol = 10^{-9}, OptimalityTol = 10^{-9}, NumericFocus = 3, Threads = 20, Seed = 0, and no finite time limit. Continue to Step 1.

Step 1. Determine the search domain. Determine $1 \le N \le N^{UB}$ using Eqs. (29)–(37). If $N^{UB} < 1$, classify the instance as infeasible and terminate. Set $N = 1$ and continue to Step 2.

Step 2. Evaluate a candidate N. For the current N, perform Steps 2.1–2.3.

Step 2.1. Construct the model. Minimize TF^a using Eq. (15) subject to Eqs. (16)–(28). For the computational formulation, enforce Eq. (16) by expressing the last batch size as $Q_N = n - \sum_{i=1}^{N-1} Q_i$. Continue to Step 2.2.

Step 2.2. Solve the model. Solve the current model using Gurobi spatial branch-and-bound. If the returned status is INF_OR_UNBD, set DualReductions = 0 and solve the same model again. Continue to Step 2.3.

Step 2.3. Validate and retain the result. If the final status is OPTIMAL, verify Q_i, $B_{k,i}$, $C_{k,i}$, and TF^a using Eqs. (15)–(17) and (20)–(28), and retain the result. If the final status is INFEASIBLE, retain the status. Otherwise retain the final solver status for complete-domain verification. Return the result for the current N.

Step 3. Continue the enumeration. If $N < N^{UB}$, set $N \leftarrow N + 1$ and return to Step 2. Otherwise continue to Step 4.

Step 4. Verify complete-domain resolution. If every $N = 1, \dots, N^{UB}$ ends with OPTIMAL or INFEASIBLE status, continue to Step 5. Otherwise classify the instance as unresolved for complete-domain global verification and end.

Step 5. Select the global reference. Among all OPTIMAL results, select the solution with the smallest TF^a using Eq. (63). Report $N^\star$, $Q_i^\star$, $B_{k,i}^\star$, $C_{k,i}^\star$, and $TF^{a\star}$.

End.

Algorithm 1 performs complete enumeration over the computational domain of N. The analytical estimate, KKT procedure, and directional neighborhood search used by the proposed heuristic are not employed in this procedure.

3.4. Numerical Illustration

The numerical illustration uses the same instance presented in the mathematical-model development. The production network consists of three sequential processing stages distributed across two companies. Stages 1 and 2 are operated by Company 1, whereas Stage 3 is operated by Company 2. The production-network configuration is shown in Fig. 1.

The computational-domain procedure in Section 2.5.1 gives seven admissible integer values for the number of batches. Therefore, seven independent nonconvex models are solved, one for each candidate number of batches. All seven candidate models are resolved under the zero-gap target, and the resulting objective values and terminal solver statuses are presented in Table 2.

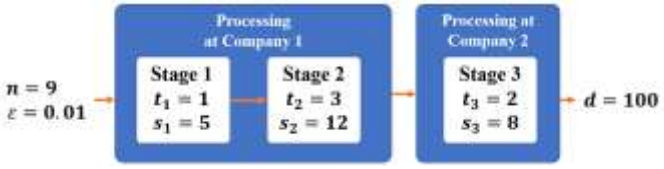

Fig. 1. Production network configuration for the numerical illustration.

Tab. 2. Gurobi results for each candidate number of batches in the numerical example

N	Minimum TF^a	Solver status
1	486.00	OPTIMAL
2	355.95	OPTIMAL
3	**343.80**	**OPTIMAL**
4	343.83	OPTIMAL
5	343.98	OPTIMAL
6	344.25	OPTIMAL
7	390.39	OPTIMAL

Among the seven solver-verified solutions, the minimum Total Actual Flow Time is obtained at $N^G = 3$, with $TF_G^a = 343.80$ and $Q_i^G = (3.00, 4.20, 1.80)$. The selected solution satisfies demand balance, the minimum batch-size requirement, nonnegative processing start times, same-shop-floor sequencing, inter-shop-floor precedence, and completion of Batch 1 at the common due date.

For the selected solution, the processing start times at Shop Floor 1 are 82.00, 56.20, and 41.20. The direct and decomposed calculations of TF^a produce the same value, confirming consistency between the mathematical formulation, backward-scheduling recursion, and objective decomposition.

3.5. Computational Evaluation of the Solver-Based Procedure

The solver-based procedure was applied to 300 computational instances covering small, medium, and large problem categories. The instance set includes conditions in which processing and setup parameters are identical across shop floors and conditions in which one or both parameter types differ. The complete dataset and parameter combinations are provided in Appendix C.

Under the final zero-gap protocol, all 300 benchmark instances were globally resolved over the complete computational domain, subject to solver numerical tolerances. For every instance, each integer value for the number of batches within the established computational domain was attempted, and the model corresponding to every candidate terminated with either OPTIMAL or INFEASIBLE status. The minimum audited objective among the OPTIMAL candidate models therefore defines the complete-domain Gurobi global reference for that instance.

Appendix D reports the grouped instance information, selected values for the number of batches, Q_i, TF^a, pairwise solution comparisons, and per-instance computational records for the three methods.

The main computational burden remains the need to solve a separate nonconvex spatial branch-and-bound model for every integer value for the number of batches in the computational domain. As the domain expands and the number of batch-size and scheduling variables increases, the complete global-reference procedure may require substantial computational effort.

4. Development of the Heuristic Method

The proposed heuristic is developed to reduce the computational effort required by complete enumeration of N with spatial branch-and-bound for every candidate. The method determines the number of batches, continuous batch sizes, and the corresponding backward schedule by combining an initial continuous estimate of the number of batches, KKT-based Q_i determination, backward-schedule reconstruction, and a directional neighborhood search.

4.1. Overall Solution Framework

The heuristic first determines the computational and analytical bounds established in Section 2.5 and obtains the continuous search center. Neighboring integer values for the number of batches are then evaluated. For each evaluated $\boldsymbol{N}$, the equal-sized partition is used only for initialization; Q_i are subsequently determined with the KKT procedure, followed by backward-schedule reconstruction and complete feasibility validation. Adjacent values of $\boldsymbol{N}$ are evaluated according to the directional search rule until its stopping condition is reached.

4.2. Continuous Estimate and Initial Candidate Generation

The total unit processing time P is defined in Equation (39). To obtain a reference setup-processing pair for the continuous estimate, the reference setup time is selected as the largest setup time among all shop floors:

$$s^c = \max_{k=1,\dots,K} s_k. \tag{64}$$

When several shop floors have the same value of s^c, the associated reference processing time is selected as

$$t^c = \max\{t_k \mid s_k = s^c\}. \tag{65}$$

The quantities P, s^c, and t^c are used only to position the initial search. They do not replace the actual t_k and s_k in the mathematical model, backward-scheduling equations, or feasibility checks. The analytical bounds $N^A_{\min}$ and $N^A_{\max}$ are used to keep the initial search center within the region satisfying the scheduling-horizon conditions under the equal-Q_i reference, while the subsequent directional search may still evaluate values outside these bounds within the computational domain. For $s^c > 0$, define the preliminary continuous estimate of the number of batches N^0 as

$$N^0 = \sqrt{\frac{(2P - t^c)n}{s^c}}. \tag{66}$$

If $s^c = 0$, N^0 is set to 1. When $N^A_{\min} \le N^A_{\max}$, the preliminary estimate N^0 is projected onto the analytical bounds to obtain the continuous search center N^c:

$$N^c = \min\{N^A_{\max}, \max(N^A_{\min}, N^0)\}. \tag{67}$$

When $N^A_{\min} > N^A_{\max}$, the analytical bounds are empty and N^0 is projected onto the complete computational domain to obtain N^c:

$$N^c = \min\{N_{UB}, \max(N_{LB}, N^0)\}. \tag{68}$$

Equations (66)-(68) provide a continuous search center rather than a closed-form optimum of the complete unequal-batch model. The initial integer candidates are

$$N_1 = \lfloor N^c \rfloor, \qquad N_2 = \lceil N^c \rceil, \qquad N_{LB} \le N_1, N_2 \le N_{UB}. \tag{69}$$

Duplicate candidates are removed. When $N_1 = N_2$, the center and its available neighboring integers are evaluated to determine the improving direction. The analytical estimate therefore initializes the search but does not exclude feasible unequal-batch solutions elsewhere within the computational domain.

4.3. Batch-Size Determination Using KKT

For an evaluated value of N, the backward-scheduling recursion in Equation (26) determines each completion time by

comparing $B_{k+1,i}$ and $B_{k,i-1} - s_k$. The two-candidate completion-time expressions are

$$D_{k,i} = B_{k+1,i}, \qquad M_{k,i} = B_{k,i-1} - s_k. \tag{70}$$

At each scheduling position, Eq. (70) compares two candidate completion-time expressions: $B_{k+1,i}$, representing downstream precedence, and $B_{k,i-1} - s_k$, representing same-shop-floor sequencing. Three scheduling conditions may therefore occur. If $B_{k+1,i} < B_{k,i-1} - s_k$, downstream precedence is active and $C_{k,i} = B_{k+1,i}$. If $B_{k+1,i} > B_{k,i-1} - s_k$, same-shop-floor sequencing is active and $C_{k,i} = B_{k,i-1} - s_k$. If $B_{k+1,i} = B_{k,i-1} - s_k$, a tie occurs; both conditions give the same completion time and both alternatives are retained for conditional evaluation. The collection of these scheduling conditions over all relevant positions defines the scheduling-condition pattern used in the conditional KKT procedure. For a given scheduling-condition pattern, the processing starts and completion times can be expressed as affine functions of the Q_i:

$$\mathbf{B}_r = A_{B,r}\mathbf{Q} + \mathbf{b}_{B,r}, \qquad \mathbf{C}_r = A_{C,r}\mathbf{Q} + \mathbf{b}_{C,r}. \tag{71}$$

Substitution into the objective in Equation (15) produces the conditional quadratic form

$$TF_r^a(N, \mathbf{Q}) = \frac{1}{2}\mathbf{Q}^T H_r \mathbf{Q} + \mathbf{c}_r^T \mathbf{Q} + \alpha_r. \tag{72}$$

The corresponding conditional problem is

$$\min_{\mathbf{Q}} \ \frac{1}{2}\mathbf{Q}^T H_r \mathbf{Q} + \mathbf{c}_r^T \mathbf{Q} + \alpha_r, \qquad \text{subject to } \mathbf{1}^T\mathbf{Q} = n,\ G_r\mathbf{Q} \geq \mathbf{h}_r. \tag{73}$$

The inequalities in Eq. (73) collect the minimum batch-size requirements, nonnegative processing start times, scheduling-feasibility conditions, and relationships required for consistency with the assumed scheduling conditions. The corresponding Lagrangian is

$$L_r = TF_r^a(N, \mathbf{Q}) + \lambda(\mathbf{1}^T\mathbf{Q} - n) - \boldsymbol{\mu}^T(G_r\mathbf{Q} - \mathbf{h}_r). \tag{74}$$

The KKT conditions are

$$\begin{aligned} &H_r\mathbf{Q} + \mathbf{c}_r + \lambda\mathbf{1} - G_r^T\boldsymbol{\mu} = 0, \\ &\mathbf{1}^T\mathbf{Q} = n, \qquad G_r\mathbf{Q} \geq \mathbf{h}_r, \qquad \boldsymbol{\mu} \geq 0, \\ &\mu_j\left(\mathbf{g}_j^T\mathbf{Q} - h_j\right) = 0 \qquad \forall j. \end{aligned} \tag{75}$$

When a batch size reaches ε, the corresponding lower-bound constraint is treated as active, and the KKT system is resolved for the remaining free batch sizes. The resulting Q_i values are used to reconstruct the complete backward schedule, and the resulting scheduling conditions are compared with the assumed conditions. If the scheduling conditions change, the conditional problem is updated and solved again. Ties are handled by retaining both alternatives for conditional evaluation.

The KKT conditions are applied to the conditional subproblem associated with a given scheduling-condition pattern. They provide stationary candidates satisfying the corresponding first-order and active-constraint conditions; they do not by themselves establish global optimality of the complete nonconvex problem for a candidate N. Each candidate is retained only after demand balance, minimum batch size, nonnegative start times, all sequencing and precedence relationships, and consistency of the scheduling conditions have been verified. If more than one valid candidate is generated for the evaluated N, the candidate with the smallest TF^a is retained. The detailed conditional derivation is provided in Appendix B, while reliability is assessed computationally in Section 5.

The KKT-based conditional procedure is not intended to exhaustively enumerate all theoretically possible scheduling-condition patterns. Instead, the scheduling conditions are identified from the reconstructed backward schedule and updated when necessary, while both alternatives are considered when a tie occurs. Only candidates satisfying the complete model constraints are retained. Accordingly, global optimality is not claimed from the KKT conditions or the conditional procedure alone. Its reliability is assessed computationally in Section 5 against the independently obtained Gurobi reference over the complete benchmark dataset.

4.4. Directional Neighborhood Search for the Number of Batches

The directional neighborhood search begins with the distinct integer candidates generated by Equation (69). When two initial candidates are available, the candidate with the smaller TF^a determines the improving direction. If the two objective values are equal, the search continues in both directions. When N^c is integer-valued, the center and its available left and right neighbors are evaluated; the search continues only in directions that do not initially worsen the objective.

For a selected search direction, the next integer is generated by

$$N_{\text{new}} = N_{\text{ref}} + \delta, \qquad \delta \in \{-1, +1\}. \tag{76}$$

The direction continues when the new candidate is feasible and

$$TF^a(N_{\text{new}}) \leq TF^a(N_{\text{ref}}). \tag{77}$$

Equality is treated as a tie and does not terminate the direction. A direction stops when the computational boundary is reached, the next candidate evaluation produces no validated feasible result, or the first strict increase occurs:

$$TF^a(N_{\text{new}}) > TF^a(N_{\text{ref}}). \tag{78}$$

The first-strict-increase rule is a heuristic stopping criterion rather than a proof that TF^a is globally unimodal with respect to N. Its purpose is to limit the number of evaluated integer candidates after the analytical estimate has positioned the search in a promising region. The analytical bounds guide but do not truncate the search: when improvement continues at an analytical boundary, the adjacent integer outside the analytical bounds is evaluated as long as it remains within Eq. (37). The final heuristic solution is the validated evaluated value of N with the smallest TF^a, together with its Q_i and reconstructed backward schedule. The effectiveness of the directional stopping rule is evaluated in Section 5.

4.5. Heuristic Algorithm

The complete heuristic logic is summarized in Algorithm 2. The algorithm integrates the discrete search over N with Q_i determination, scheduling-condition updates, backward-schedule reconstruction, and mandatory validation for every evaluated candidate N.

Algorithm 2. Proposed Heuristic Solution Procedure

Step 0. Initialize. Read K, n, d, ε, t_k, and s_k, $k = 1, \ldots, K$. Continue to Step 1.

Step 1. Determine the search bounds. Determine $1 \leq N \leq N^{UB}$ using Eqs. (29)–(37). Determine $N^A_{\min}$ and $N^A_{\max}$ using Eqs. (52)–(61). Continue to Step 2.

Step 2. Determine the search center. Determine N^c using Eqs. (64)–(68). Continue to Step 3.

Step 3. Determine the initial candidate(s). Determine $\lfloor N^c \rfloor$ and $\lceil N^c \rceil$ using Eq. (69). If $\lfloor N^c \rfloor = \lceil N^c \rceil$, retain only one candidate. Continue to Step 4.

Step 4. Evaluate a candidate N. Evaluate each candidate obtained in Step 3 by performing Steps 4.1–4.4. The same procedure is used for each adjacent N evaluated in Steps 5 and 6.

Step 4.1. Initialize Q_i. Determine the initial Q_i using Eq. (38). Continue to Step 4.2.

Step 4.2. Determine the scheduling condition. Reconstruct $B_{k,i}$ and $C_{k,i}$ using Eqs. (20)–(28). For each $C_{k,i}$, determine whether the applicable scheduling condition is downstream precedence, same-shop-floor sequencing, or a tie according to Eq. (70) by comparing $B_{k+1,i}$ and $B_{k,i-1} - s_k$. If a tie occurs, retain both alternatives. Continue to Step 4.3.

Step 4.3. Determine Q_i . For each scheduling condition obtained in Step 4.2, determine Q_i using Eqs. (71)–(75). Reconstruct $B_{k,i}$ and $C_{k,i}$ using the resulting Q_i. If the scheduling condition in Eq. (70) changes, return to Step 4.2; otherwise continue to Step 4.4.

Step 4.4. Validate the result. Verify Q_i, $B_{k,i}$, and $C_{k,i}$ using Eqs. (16)–(28). If more than one feasible result is obtained for the current N, retain the result with the smallest TF^a. If no feasible result is obtained, mark the current N as unsuccessful. Return the result for the current N.

Step 5. Determine the search direction(s). If $\lfloor N^c \rfloor \neq \lceil N^c \rceil$, compare $TF^a(\lfloor N^c \rfloor)$ and $TF^a(\lceil N^c \rceil)$. If $TF^a(\lfloor N^c \rfloor) < TF^a(\lceil N^c \rceil)$, continue the search toward smaller N. If $TF^a(\lceil N^c \rceil) < TF^a(\lfloor N^c \rfloor)$, continue the search toward larger N. If both values are equal, continue the search in both directions. If only $\lfloor N^c \rfloor$ is feasible, continue toward smaller N; if only $\lceil N^c \rceil$ is feasible, continue toward larger N. If neither is feasible, terminate the heuristic.

If N^c is an integer, evaluate $N^c - 1$ and $N^c + 1$, when they lie within $1 \leq N \leq N^{UB}$, using Step 4. Continue the search in each direction for which the adjacent N gives a smaller TF^a than $TF^a(N^c)$. If neither adjacent value improves $TF^a(N^c)$, continue to Step 7. If N^c is unsuccessful, continue from each adjacent N that gives a feasible result. Continue to Step 6.

Step 6. Perform the directional search. In each selected direction, determine the next adjacent N using Eq. (76). If the new N lies outside $1 \leq N \leq N^{UB}$, stop the search in that direction. Otherwise, evaluate the new N using Steps 4.1–4.4. If no feasible result is obtained, stop the search in that direction. If Eq. (77) holds, retain the new result and continue to the next adjacent N in the same direction. If Eq. (78) holds, stop the search in that direction. Repeat Step 6 until the search has stopped in all selected directions. Continue to Step 7.

Step 7. Select the heuristic solution. Among all feasible evaluated values of N, select the solution with the smallest TF^a. Report $N^\star$, $Q_i^\star$, $B_{k,i}^\star$, $C_{k,i}^\star$, and $TF^{a\star}$. End.

Algorithm 2 avoids complete enumeration of the computational domain by using the analytical bounds and continuous search center to position the initial search, then evaluating only adjacent candidate values for the number of batches required by the directional rule. For each evaluated N, the equal-sized reference is used only for initialization; the final continuous batch sizes are determined by the KKT-based conditional procedure and retained only after completing backward-schedule reconstruction and mandatory validation. The numerical illustration of the heuristic is presented in Section 4.6.

4.6. Numerical Illustration of the Proposed Heuristic

The numerical illustration uses the same instance as the solver-based procedure in Section 3.4, with $K = 3$, $n = 9$, $d = 100$, $\varepsilon = 0.01$, $\mathbf{t} = (1,3,2)$, and $\mathbf{s} = (5,12,8)$. Stages 1 and 2 are operated by Company 1, whereas Stage 3 is operated by Company 2. The computational domain established in Section 2.5.1 is $1 \leq N \leq 7$. The following subsections illustrate how the proposed heuristic positions the initial search, determines the continuous batch sizes, applies the directional neighborhood rule, and selects the final solution.

4.6.1. Initial Search

For this instance, the total unit processing time is $P = 1 + 3 + 2 = 6$. The largest setup time is $s^c = 12$, which occurs at Stage 2, and the associated reference processing time is therefore $t^c = 3$. Substitution into Equation (66) gives the preliminary continuous estimate

$$N^0 = \sqrt{\frac{(2P - t^c)n}{s^c}} = \sqrt{\frac{(2(6) - 3)(9)}{12}} = 2.5981.$$

The analytical range defined by the bounds in Section 2.5.3 is $1 \leq N \leq 6$. Because N^0 lies inside this interval, Equation (67) gives $N^c = 2.5981$. The initial integer candidates from Equation (69) are therefore

$$N_1 = \lfloor N^c \rfloor = 2, \qquad N_2 = \lceil N^c \rceil = 3.$$

Each candidate is evaluated using Steps 4.1–4.4 of Algorithm 2. For the two initial candidate values, the equal-sized partitions are used only to initialize the $Q_i$and reconstruct the initial backward schedules. The equal-size condition is then released, the scheduling condition at each scheduling position is identified using Eq. (70), and the continuous batch sizes are recalculated through the KKT-based conditional procedure before the complete backward schedules are reconstructed and validated.

4.6.2. Evaluation of N and Directional Neighborhood Search

For the lower initial candidate, $N = 2$, Algorithm 2 produces the $Q_i = (3.90,5.10)$ and $TF^a = 355.95$. For the upper candidate, $N = 3$, the objective decreases to $TF^a = 343.80$ with $\mathbf{Q} = (3.00,4.20,1.80)$. The improving direction is therefore toward larger N, and the next adjacent value, $N = 4$, is evaluated. Table 3 summarizes the sequence.

Tab. 3. Numbers of batches evaluated by the proposed heuristic

Evaluation order	N	Q_i	Minimum TF^a	Search interpretation

1	2	(3.90, 5.10)	355.95	Initial candidate
2	3	(3.00, 4.20, 1.80)	343.80	Improving candidate
3	4	(2.9966, 4.1967, 1.7967, 0.0100)	343.83	First strict increase; stop

The solution at $N = 4$ remains feasible, but its objective value, $TF^a = 343.83$, is higher than $TF^a = 343.80$ obtained at $N = 3$. Equation (78) therefore stops the search in the increasing direction. Among the feasible values evaluated by the heuristic, the smallest objective value is obtained at $N = 3$. In this illustration, the directional rule terminates after three evaluations; its broader reliability is assessed computationally in Section 5 rather than inferred from this single example.

4.6.3. Selected Heuristic Solution and Constraint Verification

The selected heuristic solution is $N^H = 3$, $\mathbf{Q}^H = (3.00,4.20,1.80)$, $TF_H^a = 343.80$. The batch sizes satisfy the demand balance because $3.00 + 4.20 + 1.80 = 9$, and each batch satisfies $Q_i \geq \varepsilon$. The reconstructed backward schedule satisfies the nonnegative-start, same-shop-floor sequencing, and inter-stage precedence constraints, and Batch 1 at Stage 3 completes at the common due date. For the selected solution, the processing start times at Stage 1 are $B_{1,1} = 82.00$, $B_{1,2} = 56.20$, and $B_{1,3} = 41.20$. The direct objective calculation is

$$TF_H^a = (100 - 82.00)(3.00) + (100 - 56.20)(4.20) + (100 - 41.20)(1.80) = 343.80.$$

Consistent with Proposition 1, $RBW = 0$. The decomposed objective gives the same result:

$$TF_H^a = PTF_1 + PTF_2 + PTF_3 + TWW + FGW$$
$$TF_H^a = 29.88 + 89.64 + 59.76 + 51.00 + 113.52 = 343.80.$$

Thus, the selected heuristic solution satisfies the complete mathematical constraints and produces identical direct and decomposed objective values within the adopted numerical tolerance. The corresponding backward schedule is illustrated in Fig. 2.

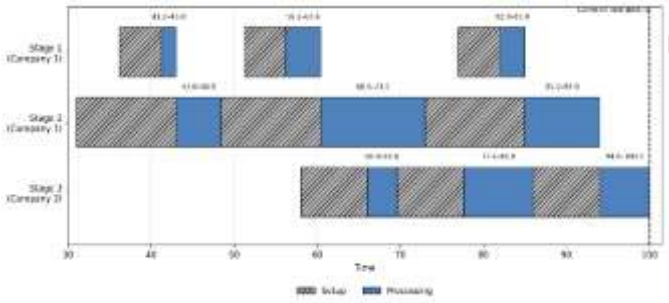

Fig. 2. Backward Gantt chart of the selected heuristic solution.

5. Numerical Experiment and Comparative Analysis

5.1. Numerical Experiment Design

The numerical experiment uses 300 instances grouped as Small 1–100, Medium 101–200, and Large 201–300. Small instances contain 2–5 processing stages and 6–9 units, medium instances contain 2–5 stages and 10–20 units, and large instances contain 6–10 stages and 10–20 units. The proposed heuristic, complete-domain Gurobi reference, and independent continuous Simulated Annealing benchmark use identical instance parameters and computational domains. Appendix D reports Q_i and TF^a.

5.2. Independent Simulated Annealing Benchmark

Simulated Annealing (SA) [56] is used solely as an independent stochastic comparator and is not part of the proposed heuristic. It does not use the analytical estimate of the number of batches, KKT-based batch sizing, directional neighborhood search, or information from Gurobi. The SA state consists of an integer number of batches and a Q_i satisfying the demand-balance and minimum-batch-size requirements. Each candidate is evaluated by reconstructing the complete backward schedule using the original model equations.

The final SA benchmark uses $T_0 = 1000, \alpha = 0.95, T_{min} = 10$, and $L = 200$ candidate moves per temperature level. Ten independent replications are performed for each instance, and the best objective value among the ten replications is used for solution-quality comparison. The complete procedure is summarized in Algorithm 3.

Algorithm 3. Independent Continuous Simulated Annealing Benchmark

Step 0. Initialize. Read K, n, d, ε, t_k, and s_k, $k = 1, \ldots, K$. Determine $1 \leq N \leq N^{UB}$ using Eqs. (29)–(37). Set $T_0 = 1000$, $\alpha = 0.95$, $T_{\min} = 10$, $L = 200$ candidate moves per temperature level, $R = 10$ independent replications, the deterministic seed base used by the implementation, and $r = 1$. Continue to Step 1.

Step 1. Start replication r and determine the initial state. Set the random seed for replication r. Scan $N_0 = 1, \ldots, N^{UB}$ in ascending order. For each N_0, determine Q_i using Eq. (38), reconstruct $B_{k,i}$ and $C_{k,i}$ using Eqs. (20)–(28), and evaluate feasibility. Use the first feasible result as the initial state. If no feasible initial state is obtained, terminate the SA benchmark. Set the current state and the replication-best state equal to the initial state, set $T = T_0$, and continue to Step 2.

Step 2. Evaluate a candidate move. For each candidate move at the current temperature, perform Steps 2.1–2.3.

Step 2.1. Determine N_{new} and $Q_{i,\text{new}}$. Draw r_N uniformly such that $0 \leq r_N < 1$ and determine $N_{\text{new}} = \lfloor r_N N^{UB} \rfloor + 1$.
If $N_{\text{new}} = 1$, set $Q_{1,\text{new}} = n$. Otherwise draw and sort $N_{\text{new}} - 1$ independent uniform random cut points satisfying $0 \leq u < 1$, use their consecutive differences to obtain $w_i \geq 0$ with $\sum_{i=1}^{N_{\text{new}}} w_i = 1$, and determine $Q_{i,\text{new}} = \varepsilon + (n - N_{\text{new}}\varepsilon)w_i$. Continue to Step 2.2.

Step 2.2. Evaluate the candidate. Reconstruct $B_{k,i}$ and $C_{k,i}$ using Eqs. (20)–(28). If the candidate is infeasible, reject it and continue to Step 3. Otherwise determine TF_{new}^a using Eq. (15), calculate $\Delta = TF_{\text{new}}^a - TF_{\text{current}}^a$, and continue to Step 2.3.

Step 2.3. Apply the Metropolis rule. If $\Delta \leq 0$, accept the candidate. If $\Delta > 0$, draw r_A

uniformly such that $0 \le r_A < 1$ and accept the candidate if $r_A < \exp\left(-\frac{\Delta}{T}\right)$.

Otherwise retain the current state. If an accepted candidate gives a smaller TF^a than the replication-best result, update the replication-best result. Continue to Step 3.

Step 3. Continue the temperature level. If fewer than L candidate moves have been evaluated at the current temperature, return to Step 2. Otherwise set $T \leftarrow \alpha T$. If $T > T_{\min}$, return to Step 2 for the next temperature level; otherwise continue to Step 4.

Step 4. Complete replication r. Record the replication-best result. If $r < R$, set $r \leftarrow r + 1$ and return to Step 1. Otherwise continue to Step 5.

Step 5. Select the SA solution. Among the R replication-best results, select the solution with the smallest TF^a. Report $N^\star$, $Q_i^\star$, $B_{k,i}^\star$, $C_{k,i}^\star$, and $TF^{a\star}$. **End.**

The SA benchmark therefore provides an independent stochastic comparison using continuous batch sizes over the same computational domain, without using the heuristic analytical search center, KKT results, directional-search information, or Gurobi solution information.

5.3. Comparative Computational Results

5.3.1. Gurobi Global Reference over the Complete 300-Instance Dataset

The final Gurobi benchmark globally resolves all 300 computational instances over the complete computational domain for the number of batches under the zero-gap protocol, subject to solver numerical tolerances. Accordingly, one complete-domain Gurobi global reference is available for every benchmark instance. Table 4 summarizes the final three-method comparison.

Tab. 4. Overall computational comparison across the complete 300-instance benchmark

Method	Solution quality	Same selected N	Mean time (s/instance)
Gurobi zero-gap global reference	300/300 globally resolved	Reference	—
Proposed heuristic	Global-reference objective agreement 300/300	100.00%	0.2349
Full Simulated Annealing	Mean objective gap 1.0464%	75.67%	15.3545

The proposed heuristic requires 0.2349 s per instance on average. Full Simulated Annealing requires approximately 1.5355 s per stochastic replication and 15.3545 s per instance for the complete ten-replication protocol. Thus, the full SA protocol requires approximately 65.36 times the mean computational time of the proposed heuristic, corresponding to a 98.47% reduction in computational time for the heuristic relative to SA. The Gurobi procedure is used primarily as the complete-domain solution-quality reference, and its per-instance computational records are reported in Appendix D rather than used as the primary aggregate efficiency comparison. The heuristic evaluates only a localized set of candidate values for the number of batches, whereas Simulated Annealing performs approximately 18,000 candidate evaluations per replication, or 180,000 candidate moves per instance across the complete stochastic protocol.

5.3.2. Proposed Heuristic VS. Gurobi Global Reference

Against the complete-domain Gurobi global reference reported in Appendix D, the proposed heuristic obtains the same selected number of batches for all 300 benchmark instances. The reported Total Actual Flow Time also agrees for all 300 instances at the adopted two-decimal reporting precision. This complete agreement is used as the primary solution-quality benchmark for the proposed heuristic.

At the adopted two-decimal reporting precision, the reported Q_i also agree between Gurobi and the proposed heuristic for all 300 benchmark instances. Thus, the selected number of batches, reported batch sizes, and reported Total Actual Flow Time are fully aligned across the complete benchmark.

5.3.3. Simulated Annealing VS. Gurobi Global Reference

Across all 300 instances, Full Simulated Annealing selects the same reported number of batches as the Gurobi global reference in 227 cases (75.67%). Its mean, median, and maximum objective gaps relative to the Gurobi global reference are 1.0464%, 0.3404%, and 7.7641%, respectively. A total of 202 instances (67.33%) are within a 1% gap, while 289 instances (96.33%) are within a 5% gap. No Simulated Annealing result provides a lower reported objective than the Gurobi global reference.

The mean objective gap increases from 0.1796% for small instances to 0.9816% for medium instances and 1.9779% for large instances. The corresponding agreement in the selected number of batches decreases from 93% to 71% and 63%, respectively. Hence, the larger stochastic search budget improves the benchmark substantially, but its performance still deteriorates as problem size increases.

5.3.4. Integrated VS. Separate Company-Based Scheduling

To examine the value of coordinating batch decisions across companies, the same numerical instance is also evaluated using two company-based separate-scheduling references. In the first scenario, Company 1 determines the number of batches and batch sizes using Stages 1-2 only; in the second scenario, Company 2 determines them using Stage 3 only. Each reference uses the same common due date. The locally selected batch structure is then retained while the complete three-stage backward schedule is reconstructed using the original processing and setup parameters. Accordingly, all approaches are compared using the same global Total Actual Flow Time definition.

The Company 1 reference selects $N = 3$ with $\mathbf{Q} = (5.40, 3.00, 0.60)$, producing a global $TF^a = 365.40$ after the complete network schedule is reconstructed. The Company 2 reference selects $N = 2$ with $\mathbf{Q} = (6.50, 2.50)$ and produces $TF^a = 389.75$. The proposed integrated scheduling considers Stages 1-3 simultaneously and selects $N = 3$ with $\mathbf{Q} = (3.00, 4.20, 1.80)$, yielding the lowest global $TF^a = 343.80$.

Relative to the Company 1 reference, the proposed integrated scheduling reduces global Total Actual Flow Time by 5.91%. Relative to the Company 2 reference, the reduction is 11.79%. These results show that a batch structure selected from the perspective of a single company may be locally attractive but need not minimize Total Actual Flow Time for the complete

production network. Simultaneous consideration of the processing and setup parameters of all stages changes both the selected batch structure and the resulting network-wide backward schedule.

5.4. Sensitivity and Structural Analysis

5.4.1. Sensitivity to Demand

A controlled demand experiment uses the numerical example while demand varies from 1 to 20 and the remaining parameters are held constant. The selected number of batches changes stepwise from one to five. A new batch is introduced only when the reduction in TF^a obtained from smaller batches exceeds the additional setup burden. The relationship between demand and the selected number of batches is illustrated in Fig. 3.

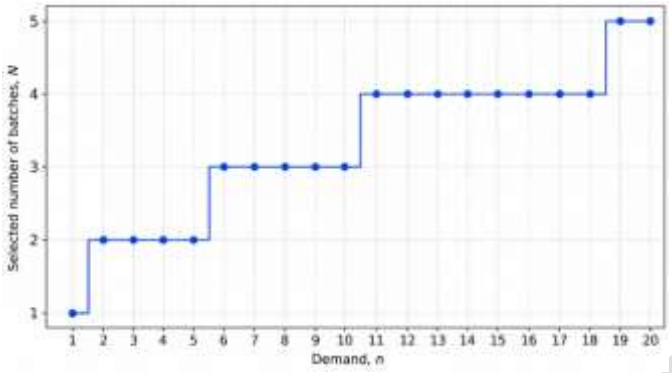

Fig. 3. Demand sensitivity of the selected number of batches.

The minimum TF^a also increases with demand because additional units increase the processing contribution and may change the batch-size distribution, active scheduling relationships, and waiting between successive shop floors. The stepwise response confirms the trade-off between smaller batches and additional setups.

5.4.2. Sensitivity to Unit Processing Times

Unit processing times are increased by 20%, 40%, 60%, 80%, and 100% using seven patterns covering changes at one shop floor, two shop floors, and all shop floors. The average results for each increase level are summarized in Table 5. The corresponding effect on the minimum Total Actual Flow Time is illustrated in Fig. 4.

Tab. 5. Summary of processing-time sensitivity

Processing-time increase	Range of selected N	Average minimum TF^a	Average runtime (s)
20%	3-4	371.82	0.03
40%	3-4	398.82	0.05
60%	4	425.49	0.04
80%	4	451.92	0.05
100%	4-5	478.10	0.06

Higher processing times increase TF^a because batches must begin earlier to meet the common due date. The location of the parameter change also matters. An increase at an intermediate processing stage affects precedence relationships on both sides, whereas an increase at the final processing stage directly increases the processing exposure near the common due date and may require earlier upstream starts.

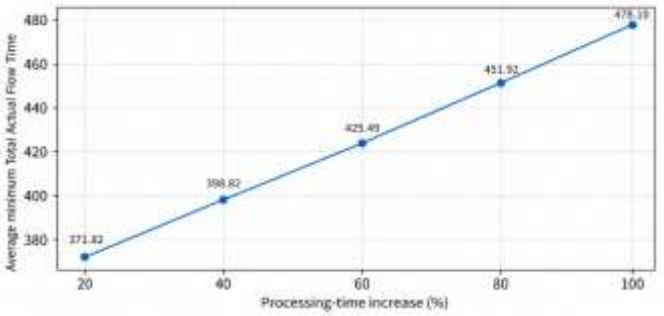

Fig. 4. Sensitivity of TF^a to processing time increases.

5.4.3. Controlled Effect of the Number of Processing Stages

A controlled sensitivity experiment is conducted to isolate the effect of the number of processing stages, K, on the batch-scheduling solution. The total demand, common due date, minimum allowable batch size, and stage processing and setup parameters are held constant, while K is varied from 2 to 10. The homogeneous processing and setup values correspond to the reference processing–setup pair used in the analytical initialization. The common due date is selected sufficiently large to retain feasible solutions over the complete tested range of K. Thus, unlike an aggregate benchmark comparison, this experiment changes only the production-network length. The results are summarized in Table 6.

Tab. 6. Controlled sensitivity to the number of processing stages

Number of processing stages, K	Selected N	Minimum TF^a	Average heuristic time (s)
2	3	348.75	0.0260
3	4	430.88	0.0489
4	4	496.13	0.0594
5	5	555.00	0.1146
6	5	604.80	0.1007
7	6	652.95	0.1399
8	6	695.25	0.1893
9	6	735.75	0.2063
10	7	774.96	0.5518

The controlled results show a clear structural effect of production-network length. As the number of processing stages increases from 2 to 10, the selected number of batches increases stepwise from 3 to 7, while the minimum TF^a obtained by the heuristic increases monotonically from 348.75 to 774.96. The increase in TF^a occurs because additional processing stages extend the production route and introduce additional precedence relationships through which each batch must pass before the common due date. The selected number of batches does not increase at every additional stage, indicating that the benefit of further batch splitting must still offset the associated setup burden. Computational time also tends to increase as K grows because each evaluation for a candidate number of batches contains more scheduling variables, precedence relationships, and scheduling positions at which the applicable condition must be evaluated. Because all parameters other than K are held constant, these results isolate the effect of production-network length within the tested design.

5.4.4. Sensitivity to the Minimum Allowable Batch Size

A controlled sensitivity experiment was conducted using the numerical illustration to examine the effect of the minimum allowable batch size, ϵ, while all other model parameters were

held constant. The experiment varies ϵ from its baseline value of 0.01 units to 3.50 units and evaluates the resulting selected number of batches and minimum Total Actual Flow Time.

For values of ε from 0.01 to 1.80 units, the optimal solution remains unchanged, with three batches, the $Q_i = (3.00, 4.20, 1.80)$, and a minimum Total Actual Flow Time of 343.80. This indicates that the minimum-batch-size constraint is inactive throughout this range because the imposed lower bound does not exceed the smallest batch size in the baseline solution. Consequently, increasing ε within this interval does not alter either the selected number of batches or the objective value.

Once ε exceeds 1.80 units, the minimum-batch-size constraint becomes binding at $\varepsilon = 00$, 2.50 , and $33.00nits$; three batches remain feasible, but the minimum Total Actual Flow Time increases to 343.95, 345.64, and 351.00, respectively. Thus, although the number of batches does not initially change, the increasingly restrictive lower bound modifies the feasible batch-size distribution and progressively increases the objective value.

A further structural change occurs at $\varepsilon = 3.50$ units. Because the total demand is nine units, three batches satisfying the lower bound would require at least $3(3.50) = 10.50$ units and are therefore infeasible. The selected number of batches consequently decreases from three to two, and the minimum Total Actual Flow Time increases to 355.95.

As illustrated in Fig. 5, the response exhibits a clear threshold effect. The objective remains constant while the minimum-batch-size constraint is inactive, begins to increase once the lower bound becomes binding, and undergoes a structural change when the restriction reduces the feasible number of batches. Hence, the effect of ε is piecewise and threshold-dependent rather than proportional across its entire range.

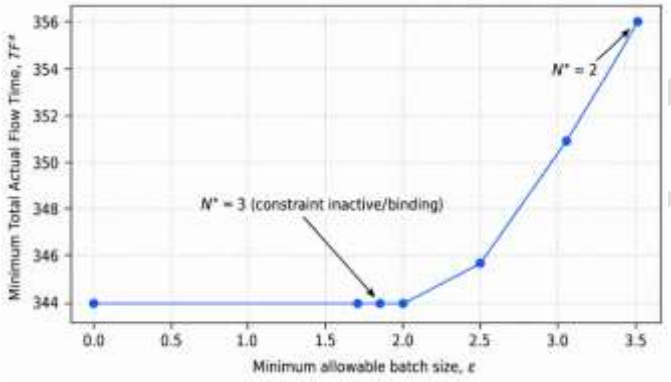

Fig. 5. Sensitivity of Total Actual Flow Time to the minimum allowable batch size.

5.5. Real-Data Application

The heuristic is further applied to a real-data instance supplied by a manufacturing company. The production network consists of seven sequential processing stages distributed across four companies. Stage 1, which involves material cutting, is performed at Company 1. Stages 2 and 3 are performed at Company 2, Stages 4–6 are performed at Company 3, and Stage 7 is performed at Company 4. The instance has a total weekly demand of 480 units and a common due date of 270,000 s. Following the order of the seven processing stages, the unit processing time is $(210, 28, 395, 187, 27, 57, 55)$ s/unit and the setup-time is $(61, 22, 37, 47, 47, 32, 10)$ s/batch. The production-network configuration is illustrated in Fig. 6.

The case-study product is a high-precision component manufactured using CNC machining. Because of its precision and handling requirements, the component is stored and handled using dedicated pallets, each accommodating 12 units. At the request of the industrial partner, the company identity, specific component designation, and product images are not disclosed.

The 12-unit pallet quantity provides the operational basis for the minimum allowable batch size used in the present model. The principal company specifies a daily delivery target to the vendor of 84 components from Monday to Friday, corresponding to seven pallets per regular working day, and 60 components on Saturday, corresponding to five pallets because of the shorter Saturday working period. Hence, the weekly demand is $5(84) + 60 = 480$ units. Because the 60-unit Saturday target occurs last in forward weekly time, it is indexed as Batch 1 under the backward batch-indexing convention. Accordingly, the company-reference Q_i is $(60, 84, 84, 84, 84, 84)$.

The physical component is discrete, and actual production, storage, handling, and delivery quantities are integer- and pallet-based. The present formulation, however, treats batch size as a continuous decision variable. Accordingly, the 12-unit pallet quantity is represented as a minimum allowable batch size rather than as an integer or pallet-multiple constraint. Thus, the present model does not generally require the resulting batch sizes to be integer-valued or multiples of 12. Incorporating integer batch-size decisions and explicit pallet-multiple restrictions is reserved for subsequent model development.

Fig. 6. Real-data production network configuration for the case-study component across four companies.

The current company policy results in six batches. Based on the computational upper-bound procedure in Section 2.5.1, the minimum-batch-size condition gives $N_\varepsilon = \lfloor 480/12 \rfloor = 40$. The scheduling-horizon bounds obtained from the individual processing stages are larger than this value; therefore, the computational upper bound is $N^{UB} = 40$, and the computational domain for the number of batches is $N = 1, \ldots, 40$. The unrestricted analytical estimate of the number of batches is 115.93. Because this value exceeds the computational upper bound, the continuous estimate used by the heuristic is restricted to the computational domain, resulting in $N^c = 40$.

Since N^c is integer-valued and coincides with the upper computational boundary, the heuristic evaluates the center $N = 40$ and its available neighboring candidate $N = 39$. The batch sizes are reoptimized for each candidate, and the corresponding backward schedules are reconstructed using the actual setup and processing times. The resulting Total Actual Flow Time is 50,289,215.24 for $N = 39$ and 50,236,560.00 for $N = 40$. Because $N = 40$ provides the lower objective value and no larger value of N is permitted by the computational domain, the selected solution is $N = 40$. For $N = 40$, the minimum-batch-size requirement and demand-balance condition imply $Q_i = 12$ units for all batches, because the total demand is exactly $40 \times 12 = 480$ units.

The selected backward schedule produces a minimum Total Actual Flow Time of 50,236,560.00 and a minimum processing start time of 72,189.00 s. The comparison between the current company policy and the selected heuristic solution is summarized in Table 7.

Tab. 7. Comparison of the company policy and the heuristic for the real-data instance

Indicator	Current company policy	Proposed heuristic algorithm
Number of batches	6	40
Batch-size structure	(60, 84, 84, 84, 84, 84)	(12,12, … ,12,12)
TF^a	71,871,660.00	50,236,560.00
Reduction from company policy	-	30.10%
Minimum processing start time	40,663.00	72,189.00

The proposed heuristic reduces Total Actual Flow Time by 21,635,100.00, or 30.10%, relative to the current company policy. The model selects 40 batches of 12 units because the pallet-derived minimum allowable batch size does not permit a smaller batch. In this particular boundary solution, each 12-unit batch also corresponds to one dedicated pallet; however, this correspondence arises from the selected solution and should not be interpreted as an integer or pallet-multiple restriction imposed by the continuous model. The resulting 30.10% reduction therefore represents the best Total Actual Flow Time obtained under the current formulation, rather than evidence that a 40-batch or one-pallet-per-batch policy would necessarily be preferred in actual operations. The corresponding sensitivity results are presented in Table 8.

Tab. 8. Sensitivity of the real-data solution to the minimum allowable batch size

Minimum batch size ε	Implied N^{UB}	Minimum TF^a	Change vs. company policy (%)	Marginal improvement of final batch (%)
80	6	74,790,000.00	−4.06	4.442
48	10	63,128,880.00	12.16	1.744
24	20	54,445,200.00	24.25	0.468
16	30	51,609,840.00	28.19	0.204
12	40	50,236,560.00	30.10	0.105

The results show the sensitivity of the real-data solution to the minimum allowable batch size while all other production-network parameters are held constant. The tested values are treated as controlled model parameters and should not be interpreted as alternative physical pallet capacities. Decreasing $\boldsymbol{\varepsilon}$ increases the feasible limit on the number of batches and progressively reduces Total Actual Flow Time. However, the marginal benefit of the final additional batch decreases substantially, from 4.442% when moving from 5 to 6 batches to only 0.105% when moving from 39 to 40 batches. Hence, although the present Total Actual Flow Time objective selects the 40-batch boundary solution, the additional flowtime benefit near this boundary is small. This result provides a quantitative basis for considering transportation, material-handling, inspection, documentation, and shipment-frequency burdens when determining an operationally acceptable number of batches.

The selected solution should nevertheless be interpreted as a model-based operational benchmark rather than an immediately implementable policy. The present formulation considers continuous batch sizes and minimizes Total Actual Flow Time, but does not explicitly incorporate transportation or delivery costs, inventory holding or storage costs, loading and unloading costs, inspection costs, or other transfer-related costs associated with increasing the number of batches. Consequently, practical implementation may require additional restrictions such as integer and pallet-multiple batch sizes, a larger minimum operational batch size, a maximum allowable number of batches, transportation and material-handling capacity, shipment frequency, and other company-specific considerations. These operational extensions are outside the scope of the present continuous model and constitute directions for subsequent development.

5.6. Discussion

The results support the value of integrating batch decisions across interconnected companies. In the numerical example, batch structures selected from the perspective of Company 1 or Company 2 produce global Total Actual Flow Time values of 365.40 and 389.75, respectively, whereas simultaneous consideration of all three stages produces 343.80. The corresponding reductions of 5.91% and 11.79% show that a locally attractive batch structure need not minimize network-wide performance.

The complete benchmark clarifies the role of the proposed heuristic. The analytical estimate positions the discrete search, the KKT procedure determines Q_i, and the directional rule limits the number of adjacent candidate values for the number of batches that must be evaluated. Across all 300 benchmark instances, the heuristic matches the reported Gurobi objective and selected number of batches while requiring only a localized search over the number of batches. This empirical agreement provides strong computational evidence for the reliability of the combined analytical and KKT-based structure without treating the heuristic stopping rule itself as a global optimality proof.

The independent Simulated Annealing benchmark provides a contrasting stochastic search mechanism. Even with a stochastic search budget of approximately 180,000 candidate moves per instance, its mean objective gap remains 1.0464%, and the same reported number of batches as the Gurobi reference is obtained in 75.67% of the instances. The complete ten-replication SA protocol requires 15.3545 s per instance on average, compared with 0.2349 s for the proposed heuristic. Thus, the proposed heuristic requires approximately 98.47% less computational time than the full SA protocol while matching the reported Gurobi objective throughout the benchmark. These results indicate that solution quality depends on both the discrete decision on the number of batches and continuous batch-size determination and support the use of analytical guidance and KKT-based batch sizing rather than relying exclusively on stochastic exploration.

The sensitivity analyses show distinct parameter effects. Higher demand and longer processing times generally increase Total Actual Flow Time and may change the selected number of batches. The controlled production-network-length experiment isolates the effect of the number of processing stages by holding the other tested parameters constant, showing a stepwise increase in the selected number of batches and a monotonic increase in Total Actual Flow Time over the tested range. The minimum-batch-size experiment exhibits a clear threshold effect: the constraint is initially inactive,

becomes binding after the smallest unconstrained batch is reached, and can eventually reduce the feasible number of batches.

5.6.1. Managerial Implications

From a managerial perspective, the results indicate that the number and sizes of production batches should not be determined independently by individual companies when their schedules are operationally interconnected. A batch structure that is locally attractive for one company may increase waiting or force earlier processing at other stages. The proposed model can therefore be used as a coordinated planning benchmark in which participating companies share processing times, setup times, the common due date, and operational batch-size restrictions before a common batch structure and backward schedule are evaluated. In the real-data setting, this framework can complement quantity-based delivery targets by explicitly incorporating the processing and setup characteristics of participating companies into the network-wide schedule evaluation.

The coordinated solution should be interpreted as a network-wide planning reference rather than an automatic implementation rule. Before operational adoption, the participating companies should evaluate transportation capacity, shipment frequency, material-handling resources, inspection and documentation requirements, integer batch-size restrictions, and other company-specific constraints. The principal managerial contribution is therefore a structured mechanism for revealing the network-wide consequences of batch decisions that would otherwise be made independently across organizational boundaries.

5.6.2. Limitations and Practical Interpretation

The real-data application should be interpreted as a model-based operational benchmark rather than an immediately implementable policy. The minimum allowable batch size of 12 units is derived from the dedicated pallet used for the high-precision component, which accommodates 12 units. Under the present Total Actual Flow Time objective, the model favors smaller batches up to this operational lower bound. Because the physical component is discrete and actual handling is pallet-based while the present model uses continuous batch sizes, integer and pallet-multiple restrictions are not explicitly represented. Transportation and delivery costs, material-handling capacity, loading and unloading effort, inspection, documentation, inventory holding or storage, and other transfer-related considerations are also not explicitly represented. Future extensions should incorporate these factors and extend the controlled sensitivity design to broader production-network structures, heterogeneous processing and setup profiles, and additional company configurations.

6. Conclusion

This study develops an integrated batch-scheduling model for sequential flow shops in a production network processing a single item under a common due date. The model simultaneously determines the number of batches, continuous batch sizes, and a feasible backward schedule while considering stage-specific processing and setup times. A finite computational domain is established from the minimum allowable batch size and scheduling-horizon conditions, while an equal-sized reference partition is used only to guide the heuristic search.

The complete-domain Gurobi benchmark globally resolves all 300 benchmark instances under the final zero-gap protocol, subject to solver numerical tolerances. The proposed heuristic obtains the same selected number of batches and reported objective value for all 300 instances while evaluating only a localized set of candidate values for the number of batches. The independent continuous Simulated Annealing benchmark provides an additional stochastic comparison, with mean and median objective gaps of 1.0464% and 0.3404%, respectively. The proposed heuristic also requires 98.47% less mean computational time than the full Simulated Annealing protocol.

The real-data application to a seven-stage production network across four companies yields a 30.10% reduction in Total Actual Flow Time relative to the current company batching policy under the adopted minimum batch-size condition. The result should be interpreted as a model-based planning benchmark rather than an immediate operational prescription. Future research should incorporate transportation and transfer costs, material-handling capacity, integer and pallet-multiple batch-size requirements, and broader production-network structures.

Acknowledgments

The authors would like to express their gratitude for the financial support from the Center for Higher Education Funding and Assessment (PPAPT), Ministry of Higher Education, Science, and Technology of the Republic of Indonesia, and the Indonesia Endowment Fund for Education (LPDP), Program of Indonesian Education Scholarship (BPI) under contract number: 2035/J5.2.3./BPI.06/10/2021. The authors also thank Universitas Jenderal Achmad Yani for the institutional support during this research, especially from the Faculty of Manufacturing Technology. The authors also acknowledge the support of the members and academic staff of the Manufacturing Systems Research Group at Institut Teknologi Bandung (ITB) as well as the constructive comments and criticism they received during this research.

Conflict of Interest

The authors declare no conflict of interest regarding the publication of this paper.

Declaration of AI Use

The authors have employed Grammarly and ChatGPT (OpenAI) to enhance language and readability, and to check, edit, and proofread this work. The authors have used these tools to review and edit the content as necessary and assume full responsibility for the final content of the manuscript, for accuracy, originality, and integrity.

References

[1] Behnamian, J., & Fatemi Ghomi, S.M.T., *"The heterogeneous multi-factory production network scheduling with adaptive communication policy and parallel machine,"* Information Sciences, Vol. 219, 2013, PP. 181-196. Doi: 10.1016/j.ins.2012.07.020.

[2] Behnamian, J., & Fatemi Ghomi, S.M.T., *"A survey of multi-factory scheduling,"* Journal of Intelligent

Manufacturing, Vol. 27, No. 1, 2016, PP. 231-249. doi: 10.1007/s10845-014-0890-y.

[3] Behnamian, J., *"Matheuristic for the decentralized factories scheduling problem,"* Applied Mathematical Modelling, Vol. 47, 2017, PP. 668-684. doi: 10.1016/j.apm.2017.02.033.

[4] Lohmer, J., & Lasch, R., *"Production planning and scheduling in multi-factory production networks: A systematic literature review,"* International Journal of Production Research, Vol. 59, No. 7, 2021, PP. 2028-2054. Doi: 10.1080/00207543.2020.1797207.

[5] Bagheri Rad, N., & Behnamian, J., *"Recent trends in distributed production network scheduling problem,"* Artificial Intelligence Review, Vol. 55, No. 4, 2022, PP. 2945-2995. Doi: 10.1007/s10462-021-10081-5.

[6] Pérez-González, P., & Framiñan, J.M., *"A review and classification of distributed permutation flowshop scheduling problems,"* European Journal of Operational Research, Vol. 312, No. 1, 2024, PP. 1-21. doi: 10.1016/j.ejor.2023.02.001.

[7] Becker, T., Neufeld, J., & Buscher, U., *"The distributed flow shop scheduling problem with inter-factory transportation,"* European Journal of Operational Research, Vol. 322, No. 1, 2025, PP. 39-55. Doi: 10.1016/j.ejor.2024.10.026.

[8] Rostami, M., & Mohammadi, M., "Two-machine decentralized flow shop scheduling problem with inter-factory batch delivery system," Operational Research, Vol. 24, No. 3, 2024, Article 39, PP. 1-37. doi: 10.1007/s12351-024-00844-7.

[9] Ziadlou, G., Emami, S., & Asadi-Gangraj, E., "Network configuration distributed production scheduling problem: A constraint programming approach," Computers & Industrial Engineering, Vol. 188, 2024, Article 109916. doi: 10.1016/j.cie.2024.109916.

[10] He, P., Zhang, B., Lu, C., Meng, L., & Zou, W.-Q., "Optimizing distributed reentrant heterogeneous hybrid flowshop batch scheduling problem: Iterative construction-local search-reconstruction algorithm," Swarm and Evolutionary Computation, Vol. 90, 2024, Article 101681. Doi: 10.1016/j.swevo.2024.101681.

[11] Hall, N.G., & Potts, C.N., *"Supply chain scheduling: Batching and delivery,"* Operations Research, Vol. 51, No. 4, 2003, PP. 566-584. Doi: 10.1287/opre.51.4.566.16106.

[12] Agnetis, A., Aloulou, M.A., & Kovalyov, M.Y., *"Integrated production scheduling and batch delivery with fixed departure times and inventory holding costs,"* International Journal of Production Research, Vol. 55, No. 20, 2017, PP. 6193-6206. Doi: 10.1080/00207543.2017.1346323.

[13] Prasetyaningsih, E., Suprayogi, Samadhi, T.M.A.A., & Halim, A.H., *"Model of integrated production and delivery batch scheduling under JIT environment to minimize inventory cost,"* Proceedings of the 2014 International Conference on Industrial Engineering and Operations Management, 2014, PP. 2109-2117. Available: https://ieomsociety.org/ieom2014/pdfs/456.pdf

[14] Prasetyaningsih, E., Suprayogi, Samadhi, T.M.A.A., & Halim, A.H., *"Production and delivery batch scheduling with multiple due dates to minimize total cost,"* Journal of Engineering and Technological Sciences, Vol. 49, No. 1, 2017, PP. 16-36. doi: 10.5614/j.eng.technol.sci.2017.49.1.2.

[15] Rahmawati, S., Masruroh, N.A., & Rifai, A.P., *"Integrated production and delivery batching for simultaneous multi-job scheduling in multi-factory environments: Modeling, linearization, and optimization,"* Production Engineering, Vol. 20, 2026, Article 4. doi: 10.1007/s11740-025-01398-z.

[16] Rahmawati, S., Masruroh, N.A., & Rifai, A.P., *"A unified MILP framework for integrated serial–parallel production batching and inter-factory delivery in multi-factory environments,"* Production Engineering Archives, Vol. 32, No. 2, 2026, PP. 198-219. doi: 10.30657/pea.2026.32.17.

[17] Karimi, N., & Davoudpour, H., *"Integrated production and delivery scheduling for a multi-factory supply chain with stage-dependent inventory holding cost,"* Computational and Applied Mathematics, Vol. 36, No. 4, 2017, PP. 1529-1544. Doi: 10.1007/s40314-016-0305-0.

[18] Goyal, S.K., *"An integrated inventory model for a single supplier–single customer problem,"* International Journal of Production Research, Vol. 15, No. 1, 1977, PP. 107-111. Doi: 10.1080/00207547708943107.

[19] Waller, M.A., Johnson, M.E., & Davis, T., *"Vendor-managed inventory in the retail supply chain,"* Journal of Business Logistics, Vol. 20, No. 1, 1999, PP. 183-203. Available: http://econspace.net/teaching/MGT-528/Waller-Johnson-Davis-VMI.pdf

[20] Salas-Navarro, K., Florez, W.F., & Cárdenas-Barrón, L.E., *"A vendor-managed inventory model for a three-layer supply chain considering exponential demand, imperfect system, and remanufacturing,"* Annals of Operations Research, Vol. 332, 2024, PP. 329-371. doi: 10.1007/s10479-023-05793-6.

[21] Dantzig, G.B., & Ramser, J.H., *"The truck dispatching problem,"* Management Science, Vol. 6, No. 1, 1959, PP. 80-91. Doi: 10.1287/mnsc.6.1.80.

[22] Khayya, E., Medarhri, I., & Zine, R., *"A survey of the vehicle routing problem and its variants: Formulations and solutions,"* Mathematical Modeling and Computing, Vol. 11, No. 1, 2024, PP. 333-343. doi: 10.23939/mmc2024.01.333.

[23] Zhang, C., Han, Y., Wang, Y., Li, J., & Gao, K., *"A distributed blocking flowshop scheduling with setup times using a multi-factory collaboration iterated greedy algorithm,"* Mathematics, Vol. 11, No. 3, 2023, P. 581. Doi: 10.3390/math11030581.

[24] Halim, A.H., & Ohta, H., *"Batch-scheduling problems through the flowshop with both receiving and delivery just in time,"* International Journal of Production Research, Vol. 31, No. 8, 1993, PP. 1943-1955. doi: 10.1080/00207549308956833.

[25] Halim, A.H., Miyazaki, S., & Ohta, H., *"Batch-scheduling problems to minimize actual flow times of parts through the shop under JIT environment,"* European Journal of Operational Research, Vol. 72, No. 3, 1994, PP. 529-544. doi: 10.1016/0377-2217(94)90421-9.

[26] Halim, A.H., Miyazaki, S., & Ohta, H., *"Lot scheduling problems of multiple items in the shop with both receiving and delivery just in time,"* Production Planning &

Control, Vol. 5, No. 2, 1994, PP. 175-184. doi: 10.1080/09537289408919484.

[27] Halim, A.H., & Yusriski, R., *"Batch scheduling for the two-stage assembly model to minimize total actual flow time,"* Proceedings of the 10th Asia Pacific Industrial Engineering and Management Systems Conference, Kitakyushu, Japan, 14-16 December 2009. Available: https://books.google.co.id/books?id=WYIBogEACAAJ

[28] Yusriski, R., Astuti, B., Sukoyo, Samadhi, T.M.A.A., & Halim, A.H., *"Integer batch scheduling problems for a single-machine to minimize total actual flow time,"* Procedia Manufacturing, Vol. 2, 2015, PP. 118-123. doi: 10.1016/j.promfg.2015.07.021.

[29] Yusriski, R., Sukoyo, Samadhi, T.M.A.A., & Halim, A.H., *"An integer batch scheduling model for a single machine with simultaneous learning and deterioration effects to minimize total actual flow time,"* IOP Conference Series: Materials Science and Engineering, Vol. 114, No. 1, 2016, P. 012073. doi: 10.1088/1757-899X/114/1/012073.

[30] Yusriski, R., Sukoyo, Samadhi, T.M.A.A., & Halim, A.H., *"An integer batch scheduling model considering learning, forgetting, and deterioration effects for a single machine to minimize total inventory holding cost,"* IOP Conference Series: Materials Science and Engineering, Vol. 319, 2018, P. 012038. doi: 10.1088/1757-899X/319/1/012038.

[31] Zahedi, Z., Rojali, & Yusriski, R., *"Stepwise optimization for model of integrated batch production and maintenance scheduling for single item processed on flow shop with two machines in JIT environment,"* Procedia Computer Science, Vol. 116, 2017, PP. 408-420. doi: 10.1016/j.procs.2017.10.081.

[32] Zahedi, Z., Salim, A., Yusriski, R., & Haris, H., *"Optimization of an integrated batch production and maintenance scheduling on flow shop with two machines,"* International Journal of Industrial Engineering Computations, Vol. 10, No. 2, 2019, PP. 225-238. Doi: 10.5267/j.ijiec.2018.7.001.

[33] Yusriski, R., Astuti, B., Ilham, M., & Zahedi, *"Integrated batch production and multiple preventive maintenance scheduling on a single machine to minimize total actual flow time,"* IOP Conference Series: Materials Science and Engineering, Vol. 598, No. 1, 2019, P. 012083. doi: 10.1088/1757-899X/598/1/012083.

[34] Hidayat, N.P.A., Cakravastia, A., Samadhi, T.M.A.A., & Halim, A.H., *"A batch scheduling model for m heterogeneous batch processor,"* International Journal of Production Research, Vol. 54, No. 4, 2016, PP. 1170-1185. doi: 10.1080/00207543.2015.1056322.

[35] Hidayat, N.P.A., Aribowo, W., & Halim, A.H., *"A single batch processor scheduling model with additional setup times to minimize total actual flowtime of parts of multiple items,"* Journal of Physics: Conference Series, Vol. 1858, No. 1, 2021, P. 012019. doi: 10.1088/1742-6596/1858/1/012019.

[36] Halim, A.H., Hidayat, N.P.A., & Aribowo, W., *"Single-item batch-scheduling model for a flow shop with m batch-processing machines to minimize total actual flow time,"* International Journal of Technology, Vol. 13, No. 4, 2022, PP. 816-826. doi: 10.14716/ijtech.v13i4.4869.

[37] Suryadhini, P.P., Sukoyo, Suprayogi, & Halim, A.H., *"A batch scheduling model for a three-stage flow shop with job and batch processors considering a sampling inspection to minimize expected total actual flow time,"* Journal of Industrial Engineering and Management, Vol. 14, No. 3, 2021, PP. 520-537. Doi: 10.3926/jiem.3438.

[38] Suryadhini, P.P., Sukoyo, Suprayogi, & Halim, A.H., *"A batch scheduling model for a three-stage flowshop with batch processor and heterogeneous job processor to minimize total actual flowtime,"* Intelligent engineering and management for Industry 4.0, Springer International Publishing, 2022, PP. 1-12. doi: 10.1007/978-3-030-94683-8_1.

[39] Maulidya, R., Suprayogi, Wangsaputra, R., & Halim, A.H., *"Batch scheduling for hybrid assembly differentiation flow shop to minimize total actual flow time,"* IOP Conference Series: Materials Science and Engineering, Vol. 319, 2018, P. 012042. doi: 10.1088/1757-899X/319/1/012042.

[40] Maulidya, R., Suprayogi, Wangsaputra, R., & Halim, A.H., *"Batch scheduling of unique and common components for a three-stage hybrid flow shop processing different product types with multiple due dates to minimize total actual flow time,"* Intelligent engineering and management for Industry 4.0, Springer, 2022, PP. 13-24. Doi: 10.1007/978-3-030-94683-8_2.

[41] Yusriski, R., Astuti, B., Biksono, D., & Wardani, T.A., *"A single machine multi-job integer batch scheduling problem with multiple due dates to minimize total actual flow time,"* Decision Science Letters, Vol. 10, No. 3, 2021, PP. 231-240. doi: 10.5267/j.dsl.2021.4.002.

[42] Yusriski, R., Nasution, A.R.K., Lukas, Wijayanti, L., & Octaviani, S., *"A two-machine flow shop batch scheduling model to minimize total actual flow time,"* Jurnal Teknik Industri, Vol. 25, No. 2, 2023, PP. 179-194. Doi: 10.9744/jti.25.2.179-194.

[43] Kurniawan, D., Yusriski, R., Isnaini, M.M., Ma'ruf, A., & Halim, A.H., *"A flow shop batch scheduling model with pre-processing and time-changing effects to minimize total actual flow time,"* Journal of Industrial Engineering and Management, Vol. 17, No. 2, 2024, PP. 542-561. doi: 10.3926/jiem.7134.

[44] Sibarani, A.A., Prabowoputra, D.M., & Melita, P., *"Modeling batch scheduling in a three-stage flow shop with multi-item on batch-job-batch processors to minimize total actual flow time,"* Industrial Engineering & Management Systems, Vol. 24, No. 2, 2025, PP. 188-201. Doi: 10.7232/iems.2025.24.2.188.

[45] Sibarani, A.A., Setyaningrum, D.T., & Waluyo, S., *"Multi-item flow shop batch scheduling with two stages dedicated machines to minimize total actual flow time,"* International Journal of Integrated Engineering, Vol. 17, No. 1, 2025, PP. 140-153. Doi: 10.30880/ijie.2025.17.01.012.

[46] Yusriski, R., Nasution, A.R.K., Isnaini, M.M., & Halim, A.H., *"A batch scheduling model on unrelated parallel machines with resource constraints and sequence-dependent setup time to minimize total actual flow time,"* Journal of Industrial Engineering and Management, Vol. 19, No. 1, 2026, PP. 58-81. doi: 10.3926/jiem.8710.

[47] Yusriski, R., Isnaini, M.M., Suprayogi, Raharno, S., Tan, K.H., & Halim, A.H., *"A prescriptive analytics heuristic*

for batch scheduling in flexible flow shops under just-in-time production," Decision Analytics Journal, Article 100740, 2026. doi: 10.1016/j.dajour.2026.100740.

[48] Rasti-Barzoki, M., Kourank Beheshti, A., & Hejazi, S.R., "Artificial Immune System for Single Machine Scheduling and Batching Problem in Supply Chain," International Journal of Industrial Engineering & Production Research, Vol. 27, No. 2, 2016, PP. 99-107. Doi: 10.22068/IJIEPR.27.2.99.

[49] Mohammadi, M., & Fatemi Ghomi, S.M.T., *"Relax and Fix Heuristics for Simultaneous Lot Sizing and Sequencing the Permutation Flow Shops with Sequence-Dependent Setups,"* International Journal of Industrial Engineering & Production Research, Vol. 21, No. 3, 2010, PP. 147-153.

[50] Ramezanian, R., Aryanezhad, M.B., & Heydari, M., *"A Mathematical Programming Model for Flow Shop Scheduling Problems for Considering Just in Time Production,"* International Journal of Industrial Engineering & Production Research, Vol. 21, No. 2, 2010, PP. 97-104.

[51] Ghafari, E., & Sahraeian, R., *"A Two-Stage Hybrid Flowshop Scheduling Problem with Serial Batching,"* International Journal of Industrial Engineering & Production Research, Vol. 25, No. 1, 2014, PP. 55-63.

[52] Çetinkaya, F.C., & Boran Yozgat, G., "Customer Order Scheduling with Job-Based Processing and Lot Streaming In A Two-Machine Flow Shop," International Journal of Industrial Engineering & Production Research, Vol. 33, No. 2, 2022, PP. 1-17. doi: 10.22068/ijiepr.33.2.11.

[53] Tawarmalani, M., & Sahinidis, N.V., Convexification and global optimization in continuous and mixed-integer nonlinear programming: Theory, algorithms, software, and applications, Kluwer Academic Publishers, 2002. doi: 10.1007/978-1-4757-3532-1.

[54] Belotti, P., Lee, J., Liberti, L., Margot, F., & Wächter, A., *"Branching and bounds tightening techniques for non-convex MINLP,"* Optimization Methods & Software, Vol. 24, No. 4-5, 2009, PP. 597-634. doi: 10.1080/10556780903087124.

[55] Gurobi Optimization, LLC, *"Gurobi Optimizer Reference Manual,"* Version 13.0, 2026. Available: *https://docs.gurobi.com/projects/optimizer/en/current/*.

[56] Kirkpatrick, S., Gelatt, C.D., Jr., & Vecchi, M.P., "Optimization by simulated annealing," Science, Vol. 220, No. 4598, 1983, PP. 671-680. doi: 10.1126/science.220.4598.671.

Appendices

The appendices supporting the analytical verification, computational transparency, and reproducibility of this study are provided separately in the associated Zenodo repository: ***https://doi.org/10.5281/zenodo.22259321***. Due to their length and technical detail, the appendices are provided outside the main manuscript to maintain its focus and readability while ensuring that complete supporting materials remain available for verification and reproducibility.

- *Appendix A. Proof of Proposition 1*
- *Appendix B. Analytical Derivations, Objective Decomposition, and KKT Formulation*
- *Appendix C. Dataset*
- *Appendix D. Complete Computational Results and Pairwise Validation*
- *Appendix E. Numerical Illustration and Computational Trace*
- *Appendix F. Source Code for the Independent Gurobi Complete-Domain Global-Reference Implementation*
- *Appendix G. Source Code for the Proposed Heuristic Implementation*
- *Appendix H. Source Code for the Independent Continuous Simulated Annealing Benchmark*
- *Appendix I. Detailed Gurobi Computational Results*
- *Appendix J. Detailed Proposed Heuristic Computational Results*
- *Appendix K. Detailed Simulated Annealing Computational Results*

The Zenodo repository contains the complete supplementary appendices, dataset, detailed computational results, and the Jupyter notebooks used to implement the independent Gurobi benchmark, the proposed heuristic, and the independent continuous Simulated Annealing benchmark.