Eng. Sci., 2025, **37**, 1789

Engineered Science
DOI: https://dx.doi.org/10.30919/es1789

Distributed Heterogeneous Hybrid Flow-Shop Scheduling under Uncertain Processing Times: A Deep Learning and Multi-Objective Evolutionary Algorithms Framework

Zhanwen Wu,[1] Weihang Dong,[1] Jinxin Wang,[2,3] Feng Zhang,[1] Zhaolong Zhu,[2,3] Xiaolei Guo[1,3] and Pingxiang Cao[1,3,*]

Abstract

With economic globalisation and the growing demand for customisation, distributed manufacturing has become a dominant production pattern, introducing considerable scheduling challenges due to diverse product types and heterogeneous machine capabilities. This study addresses the Distributed Heterogeneous Hybrid Flow-Shop Scheduling Problem (DHHFSP) under uncertain processing times. An enhanced Long Short-Term Memory (LSTM) is developed to predict Production Completion Times (PCT) by capturing temporal dependencies across varying products and machine configurations. The optimal prediction performance is achieved with 10,000 training samples, yielding a MAE of 2.176, a MAPE of 1.32%, and a coefficient of determination (R^2) of 0.959. These predictions are integrated into a multi-objective optimisation framework based on Multi-Objective Evolutionary Algorithms (MOEAs) to minimise total order earliness and tardiness and balance production line workloads. Parameter sensitivity experiments are conducted to identify effective algorithmic configurations, ensuring fair and representative comparisons. A constructive heuristic based on a novel task assignment strategy is further introduced to generate high-quality initial solutions. Experimental results on real-world cases demonstrate that the NSGA-II, combined with heuristic initialisation, achieves the most balanced trade-off between objectives. This hybrid framework provides a robust and intelligent scheduling strategy for complex manufacturing environments by combining data-driven prediction, parameter-optimised evolutionary search, and heuristic enhancement.

Keywords: Distributed hybrid flow-shop; Production completion time; Dynamic scheduling; Multi-product systems.
Received: 14 June 2025; Revised: 23 August 2025; Accepted: 01 October 2025
Article type: Research article.

1. Introduction

With the increasing volume and diversity of customer orders and product types, single-configured production centres became inadequate for handling stricter due dates and highly customised manufacturing requirements. Consequently, distributed manufacturing patterns have emerged as a common production pattern across multiple enterprises.[1,2]

A representative example can be found in the panel furniture industry, where a single customer order typically comprises multiple product types that are further divided into several production tasks.[3] These tasks are subsequently allocated across multiple factories and production lines to be processed in parallel.[4-6] While this distributed production strategy reduces processing time and accommodates the need for product customisation, it simultaneously introduces greater scheduling complexity and additional constraints that must be addressed.[7]

This multi-factory distributed scheduling model is defined as the Distributed Heterogeneous Hybrid Flow-Shop Scheduling Problem (DHHFSP). Unlike the single-centre scheduling model, *i.e.* the Hybrid Flow-Shop Scheduling Problem (HFSP), where scheduling decisions are relatively constrained, the distributed system must consider task allocation, resource assignment, and production sequencing across geographically dispersed factories with heterogeneous capabilities. Such environments are commonly found in panel furniture manufacturing, steelmaking and continuous casting, electronic assembly, and semiconductor, where production tasks are split across multiple production lines for efficiency and customisation. The complexity of the DHHFSP is further

[1]*College of Materials Science and Engineering, Nanjing Forestry University, Nanjing, Jiangsu, 210037, China*
[2]*College of Furnishings and Industrial Design, Nanjing Forestry University, Nanjing, Jiangsu, 210037, China*

compounded by stochastic processing times, machine heterogeneity, fluctuating availability, all of which can significantly impact overall production efficiency and reliability.[8,9]

The complexity of the DHHFSP arises from several key factors:

1. **Uncertain Processing Times:** Variability in factory layouts, production line configurations, and machine capabilities leads to different processing cycles times for different product types, thereby complicating scheduling decisions.
2. **Workload Balancing:** Ensuring an equitable workload across factories to prevent procedure bottlenecks and maintain a steady production flow.
3. **Meeting Order Due Date and Constraints:** The simultaneous processing of orders by multiple production lines brings uncertainties in overall order completion times and needs to minimise tardiness and earliness.

As aforementioned challenges, the most pivotal aspect of solving the DHHFSP under uncertain processing times is the accurate estimation of Production Completion Time (PCT), it directly impacts task allocation, workload distribution, and avoids delays.[10] However, existing analytical models and mathematical simulation methods often fail to capture the complex, nonlinear, and stochastic dynamics of real-world manufacturing systems. As a result, scholars have increasingly turned to data-driven approaches for more reliable PCT prediction. Among these, deep learning has shown particular promise. In this context, LSTM networks are especially suitable because they can effectively capture long-range temporal dependencies and sequential patterns in production processes,[11] LSTM can learn from heterogeneous and time-varying data, making them well equipped to represent the dynamic interactions between machines, products, and processing sequences. This capability makes LSTM an ideal choice for modelling PCT under uncertainty in distributed hybrid flow-shop environments.[12]

For the DHHFSP, several studies have explored resolve approaches for this complex scheduling model. Among them, Multi-Objective Evolutionary Algorithms (MOEAs) have emerged as the most common optimisation approach due to their ability to efficiently search the solution space and obtain high-quality solutions.[13] Scholars adapted various MOEAs variants, such as NSGA-II, MOPSO, and SPEA2, to balance multiple conflicting objectives, such as minimising completion time and energy consumption, demonstrating their effectiveness in solving large-scale and computationally challenging DHHFSP instances.

While prior research has shown that LSTM networks are effective in predicting processing times, few studies have incorporated such predictive models into dynamic distributed hybrid flow-shop scheduling. This represents a notable gap, as accurate time prediction could greatly improve production efficiency and responsiveness under uncertainty. Although the DHHFSP has been investigated in previous works, most approaches either assume deterministic processing times or employ oversimplified predictive models, which fail to capture the nonlinear and stochastic characteristics of real manufacturing systems. In particular, the integration of advanced prediction methods with optimisation strategies in the context of DHHFSP has received very limited attention.

This paper addresses the DHHFSP under uncertain processing times by proposing a novel hybrid framework that integrates LSTM networks with MOEAs. The framework can accurately predict cycle times for diverse product types across heterogeneous production lines and efficiently search for optimal scheduling solutions under multiple constraints. The main contributions of this work are as follows: 1) Developed an enhanced LSTM network that models temporal relationships across product types and machine capabilities, enabling accurate prediction of PCT. 2) Proposed a constructive heuristic method to generate high-quality initial solutions, thereby improving the convergence and performance of MOEAs in complex scheduling scenarios. 3) Validated the effectiveness of the framework through extensive benchmark experiments and real-world data from a panel furniture manufacturing enterprise.4) Demonstrated improvements in prediction accuracy, scheduling efficiency, and decision-making compared with existing approaches.

The remainder of this paper is structured as follows. Section 2 reviews related work on solving DHHFSP and PCT prediction. Section 3 details LSTM network modelling and DHHFSP modelling. Section 4 details the solution methodology of MOEAs. Section 5 assesses and validates the effectiveness of the proposed approach. Conclusions of this work will be found in Section 6.

2. Related works

2.1 Distributed heterogeneous hybrid flow-shop scheduling problem

The HFSP has received extensive attention in the operations research and manufacturing communities due to its practical relevance and computational complexity. HFSP involves scheduling a set of jobs across multiple sequential stages,

[3]*Co-Innovation Center of Efficient Processing and Utilization of Forest Resources, Nanjing Forestry University, Nanjing, Jiangsu, 210037, China*
**Email:* njfucpx@163.com (Pingxiang Cao)

where each stage may consist of several parallel machines, allowing alternative routing options. It offers enhanced flexibility in load balancing and resource utilisation, helping to alleviate production bottlenecks and improve throughput.[14] This scheduling framework is widely applied in various industries, including semiconductor fabrication, electronics assembly, textile processing, paper production, and steel rolling. However, HFSP is proven to be NP-hard, especially in instances involving multiple stages and resource constraints. This computational complexity presents significant challenges for finding optimal or near-optimal solutions within reasonable timeframes,[15] thereby motivating the development of advanced optimisation algorithms and heuristic approaches to tackle large-scale, real-world scheduling problems.

The DHHFSP, as a complex extension of the classical HFSP, introduces additional layers of structural and dynamic complexity. Unlike standard HFSP, which focuses on sequencing jobs within a single facility, DHHFSP involves simultaneous job allocation across multiple geographically distributed production sites, each equipped with heterogeneous production lines featuring distinct capabilities, configurations, and resource constraints. This requires not only determining the optimal job sequence at each stage, but also assigning jobs to suitable facilities while considering inter-factory coordination, transportation delays, and system-wide workload balancing. Moreover, stochastic factors such as uncertain processing times, fluctuating job priorities further exacerbate the scheduling challenge, rendering conventional deterministic models insufficient.[16] As a result, solving DHHFSP necessitates advanced modelling techniques and robust optimisation frameworks capable of adapting to uncertainty and heterogeneity in real-world production environments.

Several studies have explored different variants of the DHHFSP. Shao *et al*.[17] focus on the Distributed Heterogeneous Hybrid Blocking Flow-Shop Scheduling Problem (DHHFSP-B), they introduced a Learning-Based Selection Hyper-Heuristic (LS-HH) that combines a learning probability model for heuristic selection with a simulated annealing-based acceptance mechanism to improve optimisation performance. In a related study, Li *et al*.[16] addressed DHHFSP under variable speed constraints and proposed a Knowledge-Based Adaptive Reference Points Multi-Objective Algorithm (KMOEA). This approach incorporates efficient initialisation, a Pareto-based crossover heuristic, and a local search mechanism for speed adjustment. Zhao *et al*.[18] investigated the Energy-Aware Distributed Heterogeneous Flexible Flow-Shop Scheduling Problem (EADHFFSP) with variable speed constraints. They proposed a Co-Evolutionary Algorithm with a Dueling Reinforcement Learning Mechanism (DRLCEA), which integrates knowledge-based initialisation, adversarial generative learning for global search, and dueling DDQN for local search operator selection. Pan *et al*.[19] addressed the Distributed Energy-Efficient Parallel Machines Scheduling Problem (DEPMSP) and developed a Knowledge-Based Two-Population Optimisation (KTPO) algorithm to combines problem-specific heuristics with NSGA-II and differential evolution, supported by knowledge-based local search operators.

Beyond methodological improvements, several studies have extended the DHHFSP framework to specific industrial contexts and additional problem settings. Cui *et al*.[13] explored the application of DHHFSP in a global manufacturing context. They proposed a greedy job insertion inter-factory neighbourhood structure alongside a novel move evaluation method. They developed an Improved Multi-Population Genetic Algorithm (IMPGA) based on this structure, demonstrating successful application in polyester film manufacturing. In another extension of DHHFSP, Shao *et al*.[7] investigated the problem under non-identical time-of-use electricity tariffs (DHHFSP-NTOU). They introduced an Ant Colony Optimisation-based Multi-Objective Evolutionary Algorithm (ACO_MOEA/D). Zhao *et al*.[20] addressed the Distributed Heterogeneous No-Wait Flow-Shop Scheduling Problem with Sequence-Dependent Setup Times (DHNWFSP-SDST), considering factory configuration differences and transportation times. They proposed a Q-learning-driven Artificial Bee Colony Algorithm (QABC), which was enhanced by heuristic-based initialisation and adaptive neighbourhood selection. Their findings highlight the algorithm's efficiency in generating high-quality schedules, offering a promising approach for large-scale industrial applications.

However, a review of the existing literature reveals several limitations. Most studies on the DHHFSP primarily concentrate on makespan minimisation, with only a few extending to additional objectives such as energy consumption. In contrast, objectives such as total earliness and tardiness which commonly addressed in the broader HFSP literature have received little attention in the distributed setting. More importantly, the critical challenge of workload balancing across multiple factories or production lines remains largely unexplored, despite its direct impact on order completion times and overall system efficiency.

In addition, most existing works generate test instances by assuming that processing times follow a uniform distribution within a predefined range (*e.g.*, [1, 99]).[7,13,17] While such

assumptions provide a convenient testing environment, they fail to reflect the nonlinear and stochastic nature of real-world manufacturing. This oversimplification limits the robustness and practical applicability of current DHHFSP models. These gaps highlight the need for approaches that can jointly address multiple scheduling objectives—including earliness/tardiness and workload balance—while incorporating realistic, uncertainty-aware modelling of processing times. Our study directly responds to this need by integrating advanced prediction and optimisation techniques into the DHHFSP framework.

2.2 Production completion time prediction

Accurate prediction of PCT is a fundamental prerequisite for effective production scheduling, as it directly influences job sequencing, resource allocation, and due date adherence.[11] However, the increasing complexity of modern manufacturing systems, which are characterised by high product variability, dynamic shop-floor conditions, heterogeneous machine configurations, and uncertain processing behaviours, poses significant challenges to traditional estimation methods.[12] In such environments, deterministic or static prediction approaches often fail to capture the nonlinear and temporal dependencies inherent in real-time production data. As a result, there is a growing need for data-driven predictive models capable of learning from historical patterns and adapting to fluctuating operational conditions to improve the accuracy and robustness of PCT forecasting.

Early research on PCT prediction primarily relied on statistical and analytical approaches, including linear regression models, time series forecasting techniques, and queuing theory-based estimations. These methods typically assume that processing times adhere to stable and predictable probability distributions, enabling the use of historical data to infer the expected PCT of incoming jobs. While such approaches offer interpretability and are computationally efficient, they often oversimplify the dynamic and nonlinear nature of real-world manufacturing systems. Consequently, their predictive accuracy tends to degrade in complex production settings involving high variability, machine heterogeneity, or frequently changing workloads.

Building on this foundation, more advanced approaches have incorporated process mining to enhance prediction accuracy. For example, Ruschel *et al*.[21] tackled the problem of predicting task PCT while identifying performance-impacting factors in manufacturing. They proposed a framework integrating process mining techniques with probabilistic models in Bayesian networks and predictive analytics. This approach accurately calculates activity occurrence probabilities and forecasts future process behaviour. Aalst *et al*.[22] explored the application of process mining for predicting PCT by automatically deriving process models from event logs and augmenting them with temporal insights. They introduced a configurable approach to constructing and refining process models with learned time information, ensuring reliable predictions while reducing the risks of overfitting and under-fitting.

In addition to simulation-based processing time prediction, recent research has increasingly applied deep learning networks to enhance predictive accuracy. This includes: Yamashiro *et al*.[23] address the estimating unknown PCT in manufacturing. The authors proposed a machine-learning-based estimation system rather than assuming fixed or simplistic distributions. They evaluated multiple predictive models and determining that LightGBM and Ridge's regression outperform traditional probabilistic methods, reducing makespan by an average of 30%. Yuan *et al*.[11] tackle predicting PCT in a flexible job shop problem (FJSP) environment. They developed a PCT prediction model based on AE-CNN-LSTM. Feature extraction uses a multi-channel Convolutional Neural Network (CNN) and an Autoencoder (AE), while LSTM integrates real-time and historical data for precise time estimation. Simulation results confirm that the model effectively captures job shop state features, achieving over 90% prediction accuracy across varying scenarios. Huang *et al*.[12] focused on predicting PCT in complex, multi-product production systems. They proposed a hybrid approach integrating machine learning with an analytical system model to improve prediction accuracy. A mathematical model establishes a strict lower bound for completion time under the assumption of no random downtime. At the same time, a LSTM neural network predicts the deviation between this bound and actual completion times. Wang *et al*.[24] investigated adaptive scheduling in assembly job shops with uncertain processing times. They proposed a dual Q-learning method that integrates local-level strategies for machine load balancing with global-level optimisation of job earliness. Experimental results demonstrated that this approach achieved better adaptability and overall scheduling performance compared with single Q-learning and traditional dispatching rules.

Despite notable progress in PCT prediction, several critical gaps remain. Traditional statistical approaches, though effective on well-structured historical datasets, fail to capture the dynamic, nonlinear, and stochastic characteristics of modern manufacturing systems. The growing prevalence of mass customisation, machine heterogeneity, and real-time disturbances further limits the adaptability of such

conventional models. Deep learning techniques, which particularly recurrent neural networks capable of modelling temporal dependencies, have shown promise in addressing these challenges. However, prior applications have largely been confined to narrow contexts, such as standard flow-shop environments, with limited consideration of distributed and heterogeneous systems.

In this regard, a clear opportunity exists to advance the field by extending predictive modelling to the DHHFSP under uncertain processing times. Unlike existing studies, this work integrates an enhanced LSTM network with multi-objective optimisation, thereby enabling accurate, uncertainty-aware PCT forecasting and its direct utilisation in scheduling decisions. This integrated framework not only bridges the gap between prediction and optimisation but also offers a more robust and generalisable solution for complex, real-world manufacturing environments.

3. Modelling of LSTM and DHHFSP in panel furniture manufacturing

The main structure of this work is given in Fig. 1. This section is structured into two parts. The first part establishes a predictive model for PCT using the LSTM network, a prominent variant of Recurrent Neural Networks (RNNs). This model provides a mathematical framework to capture the real-time dynamics of various product types across heterogeneous production lines. The second part focuses on modelling the DHHFSP within furniture manufacturing, addressing a scheduling environment feature with multi-production line multi-stage environment.

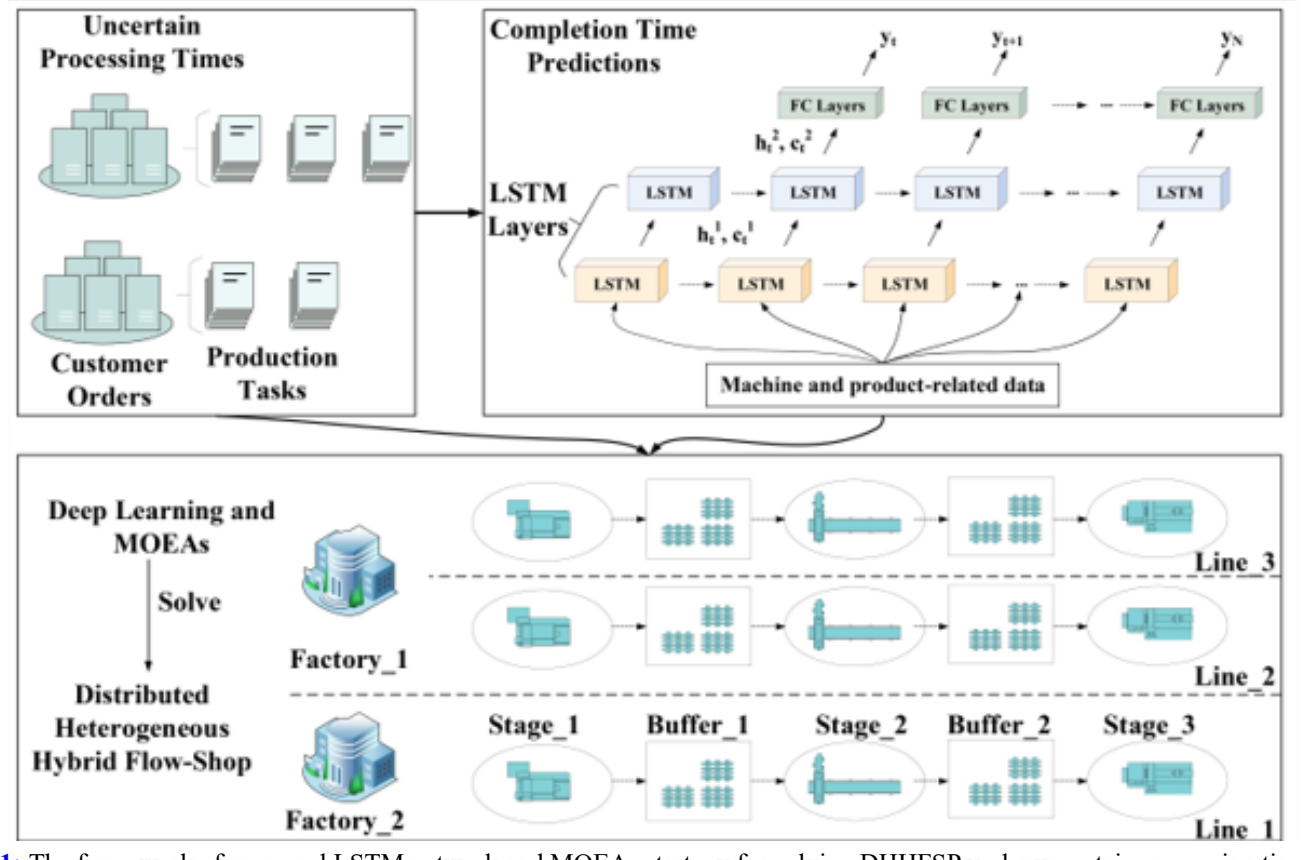


Fig. 1: The framework of proposed LSTM network and MOEAs strategy for solving DHHFSP under uncertain processing times.

3.1 System and problem description of LSTM

3.1.1 Production line system and problem formulation

This section discusses the prediction of PCT using the LSTM network for multi-product types in heterogeneous production lines.

In a manufacturing enterprise, production lines are typically heterogeneous rather than uniform, denoted as L_i. Each production line consists of M machines/workstations, denoted as M_i, and between two stations, there are M-1 buffers, denoted as B_i.

The following assumptions are made in this paper:

1. The production line can process K in different types of products. Each type of product is represented by a $K\times1$ vector q (see Eq. 1).

$$q(k)=\begin{cases}1, & \text{if the product is of the type } k\\ 0, & \text{otherwise}\end{cases} \quad (1)$$

2. Each machine/workstation can only process one product at one time.
3. As per assumption (2), the products follow a defined

sequence across all machines and cannot be altered once production enters the production line. As seen in Eq. 2, the product sequence is denoted by Q, where q_n represents the n^{th} product in the sequence.

$$Q = [q_1, q_2, \ldots, q_n, \ldots] \quad (2)$$

4. Each machine/workstation has a fixed cycle time when processing a particular product type. The cycle time T_{ik} for a machine M_i varies depending on the product type k being processed. as shown in Eq. 3.

$$Ti = [T_{i1}, T_{i2}, \ldots, T_{ik}] \quad (3)$$

5. The capacities of intermediate buffers are finite. The buffering time T_{Bi} of buffer B_i is determined by the production rhythm of the adjacent machines and falls within a fixed range, denoted by the lower bound $\bar{T}_{Bi}$ and upper bound $\hat{T}_{Bi}$ as defined in Eq. 4.

$$T_{Bi} \in [\bar{T}_{Bi}, \hat{T}_{Bi}] \quad (4)$$

3.1.2 Definition of production completion time

The PCT of product q_n refers to the time required to complete all the processing steps in the production line. The formal definition of PCT is as follows:

Definition 1: PCT of a Product

The PCT of the n^{th} product q_n in a product sequence Q at time t is denoted as $y_n(t)$ and is defined as in Eq. 5.

$$y_n(t) = max\{0, inf\{t' \in \mathbb{R}^+ Y_{Mi}(t') \geq n\} - t\} \quad (5)$$

where $Y_M(t')$ represents the cumulative production count of the final machine M_i at time t'.

In this definition, the condition $Y_{Mi}(t') \geq n$ indicates that the cumulative production count of the final machine (*i.e.*, the end of the production line) has reached or exceeded the index n at time t'. The term $inf\{t' \in \mathbb{R}^+ Y_{Mi}(t') \geq n$ represents the exact time point when product q_n exits the production line. Thus, the PCT of q_n, starts from the current time t, as defined by the above equation.

Definition 2: Lower Bound of PCT

A lower bound $\bar{y}_n(t)$ for the PCT of product q_n at time t is given by Eq. 6.

$$\bar{y}_n(t) = max\{0, inf\{t' \in \mathbb{R}^+ Y_{Mi}(t'; w(t'') = 0, \forall t'' > t) \geq n\} - t\} \quad (6)$$

where $Y_{Mi}(t'; w(t'') = 0, \forall t'' > t$ represents the cumulative production count of the final machine M_i at time t', under the condition that no downtime events occur from time t onward, *i.e.*,$w(t'') = 0, \forall t'' > t$.

Definition 3: Independence of PCT

Under the no-blocking and finite-buffer assumptions laid out, the PCT of product q_n is independent of the PCT of any product entering the Q after q_n. Mathematically, this can be expressed as in Eq. 7.

$$p(y_n(t)|y_n'(t)) = p(y_n(t)), \forall n' > n, \forall t \geq 0 \quad (7)$$

This definition states that the PCT of q_n is influenced only by its preceding products and is unaffected by any products that enter the production line after it.

3.1.3 Long short-term memory network

Traditional feed-forward neural networks are inherently limited in their ability to capture sequential dependencies, as they process input data in isolation without accounting for temporal context. This constraint renders them unsuitable for modelling time-dependent processes such as those found in manufacturing systems, where historical states and previous events critically influence current outcomes. To overcome this limitation, RNNs were introduced. RNNs are explicitly designed to handle sequential and time-series data, maintaining internal memory through recurrent connections that allow information to persist across time steps. This architecture makes RNNs particularly effective for capturing temporal patterns and modelling dynamic processes where past inputs affect future predictions.[25] It employs recurrent connections to update hidden states recursively. Given a sequence $x = [x_1, x_2, ..., x_n]$, an RNN assumes that the output at each step depends on the previous steps in the sequence. The hidden state h_n at step n is computes as in Eq. 8.

$$h_n = \sigma(W_n h_{n-1}; W_x x_n; \theta_1), \quad (8)$$

where σ represents the activation function, W_n and W_x are weight matrices, x_n is the input at time step n, and θ_l denotes the model parameters.

However, standard RNNs suffer from vanishing and exploding gradient problems. To overcome these issues, a widely used variant of RNNs, known as LSTM, is employed in this study. LSTM maintain the same chain-like structure as traditional RNNs but introduces specialised gating mechanisms to regulate information flow. Each LSTM cell has an input gate determining how much new information from x_n is stored in the cell state. A forget gate discards irrelevant information from the previous cell state, and an output gate controls the output generation of the hidden state. These mechanisms enable LSTMs to effectively capture long-term dependencies, making them well-suited for predicting PCT in hybrid flow-shop production environments.[11]

3.1.4 A Hybrid approach to production completion time prediction

This section introduces a hybrid framework for predicting PCT, integrating product-type-specific temporal modelling with an enhanced LSTM network. As outlined in the previous sections, the PCT for a given product, q_n, is influenced by its inherent characteristics and the associated machinery, particularly its product type and machine cycle time. To predict the PCT, a recurrent sequence is constructed, commencing from the final machine on the production line, denoted as M_i, and propagating backwards along the machines.

The sequence aims to predict the completion times for the incoming products, represented as [$q_{n1}(t)$, $q_{n1}(t)+1$, $q_{n1}(t)+2$, ...,

$q_{n1}(t)+N]$, where each element corresponds to a future time step.

For the product q_n, the cycle time at each machine contributes to the final PCT. Let δ_n denote the processing time at each machine, defined as in Eq. 9.

$$\delta_n = \begin{bmatrix} q_n T_1 + q_n T_{B1} \\ q_n T_2 + q_n T_{B2} \\ \cdots \\ q_n T_{M-1} + q_n T_{B_{M-1}} \\ q_n T_M \end{bmatrix} \quad (9)$$

Consequently, the input sequence x to the LSTM network is constructed by arranging the machine and product-related data, as shown in Eq. 10.

$$x = [\delta_{n1}(t), \delta_{n1}(t)_{+1}, \ldots, \delta_{n1}(t)_{+N}] \quad (10)$$

where N represents the number of incoming products.

This work employs a stacked LSTM architecture to enhance predictive accuracy. In this framework, the stacked LSTM is invoked throughout the prediction process of PCT as the default network structure. Specifically, two LSTM layers are arranged sequentially: the hidden state and cell state from the first layer are passed to the second layer, allowing the network to capture both short-term and long-term temporal dependencies. The output from the second layer is then fed into a fully connected (FC) layer to produce the final PCT prediction. Compared with a single-layer LSTM, the stacked structure introduces additional parameters and slightly increases training time. However, this extra computational cost is modest and acceptable given current resources, while the improvement in prediction accuracy is significant, as also highlighted in recent LSTM review studies.[25]

Specifically, the LSTM computations are as follows:

For the first layer of the LSTM, the formulation is given in Eq. 11.

$$h_t{}^1, c_t{}^1 = LSTM^1(x_t, h_{t-1}^1, c_{t-1}^1) \quad (11)$$

For the second layer of the LSTM, the formulation is given in Eq. 12.

$$h_t{}^2, c_t{}^2 = LSTM^2(x_t, h_{t-1}^2, c_{t-1}^2) \quad (12)$$

where h_t^l and c_t^l represent the hidden state and cell state at layer l, respectively. The output from each layer, h_t^l, is then passed as input to the subsequent layer, continuing this process through the network until the final layer.

At the final layer of the stacked LSTM, the output is passed to a fully connected (FC) layer, which performs regression through a linear transformation to generate the final PCT prediction. The computation at this layer is given by Eq. 13.

$$y_t = W h_t^2 + b \quad (13)$$

where W represents the weight matrix, b is the bias term, and h_t^2 is the output from the second layer.

To evaluate the model's performance, the Mean Squared Error (MSE) is employed as the loss function to compute the discrepancy between the predicted and actual PCT values at each time step, as defined in Eq. 14.

$$L = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \quad (14)$$

where y_i denotes the actual PCT value, $\hat{y}_i$ is the predicted PCT value, and N is the number of samples.

3.2 Mathematical modelling of the DHHFSP

3.2.1 Problem description

The DHHFSP in panel furniture manufacturing can be defined as follows. A set of N orders, denoted as $\{O_1, O_2, \ldots, O_n\}$, consist of K different type products. Based on the number of distinct product types in each order, an order O_n is divided into j production tasks $U_j^{O_n}$, such as: $O_n = \left\{U_1^{O_n}, U_2^{O_n}, \ldots, U_j^{O_n}\right\}, j \le K$. These production tasks are allocated to a set of f factories, represented as $F = \{1,2,\ldots,f\}$. Each factories F_f consist of l production line, denoted as: $F_f = \left\{L_1^{F_f}, L_2^{F_f}, \ldots, L_l^{F_f}\right\}$. Each production line follows a hybrid flow-shop layout, consisting of m machines across s processing stages, expressed as: $L_l^{F_f} = \left\{L_{l,M_1}^{F_f}, L_{l,M_2}^{F_f}, \cdots, L_{l,M_s}^{F_f}\right\}$. Additionally, limited buffers exits between stages, represented as: $\left\{L_{l,B_1}^{F_f}, L_{l,B_2}^{F_f}, \cdots, L_{l,B_b}^{F_f}\right\}, b < s$. The process sequence of each task, denoted as $Q(U_j^{O_n})$, remains consistent across all stages.

Due to heterogeneous factory layout, production lines exhibit different cycle times, denoted as $T(L_{l,M_s}^{F_f})$, and buffering times, represented as $T(L_{l,B_b}^{F_f})$. Furthermore, cycle times vary across product types, expressed as $T(L_{l,M_s}^{F_f}, k)$. When switching between different product types, a set-up time is required for tool change and other preparation work, given by $T(k^{st}, k^{now})$. Each task can be assigned to any production line; however, once allocated, all processing must be completed in that line. Tasks cannot be transferred to another line or factory after assignment.

The model incorporates fundamental assumptions of classical flow-shop scheduling.[17] Each task is processed by one machine, and each machine can handle only one task at one time. Preemption is not allowed. Once a task starts in a machine, it must be completed without interruption. All tasks and machines are available from time zero. Machine breakdowns are not considered in this study.

The purpose is to optimise both the assignment of tasks to the production lines and the processing sequence of tasks to achieve maximum workload balance across factories and minimum order earliness and tardiness. This study assumes that all factories belong to a collective, which means that certain factories can make sacrifices to maximise collective interests.

3.2.2 Mathematical formulation

Definition	Explanation
Indices:	
N	Total number of orders.
K	Total number of product types.
n	The index of orders.
j	The index of tasks.
s	The index of stages.
b	The index of buffers.
f	The index of factories.
l	The index of production lines.
Parameters:	
$U_j^{O_n}$	The j-th task of order O_n, $j \le K$, $n = 1,2,\cdots,N$.
$Q(U_j^{O_n})$	The production sequence of the j-th task of order O_n
$L_l^{F_f}$	The l-th production line of factory F_f.
$L_{l,B_b}^{F_f}$	The b-th buffer of the l-th production line of factory F_f.
$L_{l,M_s}^{F_f}$	The s-th stage machine of the l-th production line of factory F_f.
$T(L_{l,B_b}^{F_f})$	The cycle time at the b-th buffer of production line l in factory F_f.
$T(L_{l,M_s}^{F_f})$	The cycle time at the s-th stage machine of production line l in factory F_f.
$T(L_{l,M_s}^{F_f},k)$	The cycle time of product type k at the s-th stage machine of production line l in factory F_f.
$T(k^{st},k^{now})$	The set up time required when switch from product type k^{st} to k^{now}.
$P(U_j^{O_n},L_l^{F_f})$	The processing time of task j from order O_n in production line l of factory F_f.
$S(U_j^{O_n},L_l^{F_f})$	The processing start time of task j from order O_n in production line l of factory F_f..
$C(U_j^{O_n},L_l^{F_f})$	The processing completion time of task j from order O_n in production line l of factory F_f.
$E(O_n)$	The earliness of order O_n.
$T(O_n)$	The tardiness of order O_n.
$\zeta(F_f)$	The workload balance of factory F_f.
Decision variable:	
$X_{(U_j^{O_n},L_l^{F_f})}$	A 0/1 variable, which equals to 1 if task j of order O_n is processing in production line l of factory F_f.

The following model translates the problem assumptions into a mathematical formulation using the nomenclature defined earlier.

Objectives:

$$Z = \left(Max \sum_{f}^{F} \zeta(F_f), Min \sum_{n=1}^{N} E(O_n) + T(O_n) \right) \quad (15)$$

Subject to:

$$C\,max \ge C(U_j^{O_n}, L_l^{F_f}), \forall j,n,l,f \quad (16)$$

$$S(U_j^{O_n}) \ge 0, \forall j,n,l,f \quad (17)$$

$$C(U_j^{O_n}) \ge 0, \forall j,n,l,f \quad (18)$$

$$C(U_j^{O_n}, L_l^{F_f}) \ge S(U_j^{O_n}, L_l^{F_f}), \forall j,n,l,f \quad (19)$$

$$C(U_j^{O_n}, L_l^{F_f}) \le S(U_{j+1}^{O_n}, L_l^{F_f}), \forall n,l,f, 1 \le j \le J \quad (20)$$

$$S(U_{j+1}^{O_n}, L_l^{F_f}) - C(U_j^{O_n}, L_l^{F_f}) + M_s \times (1 - X_{(U_j^{O_n}, L_l^{F_f})}) \ge 0, \forall s \in S, j,n,l,f \quad (21)$$

$$S(U_j^{O_n}, L_l^{F_f}) + P(U_j^{O_n}, L_l^{F_f}) = C(U_j^{O_n}, L_l^{F_f}), \forall j,n,l,f \quad (22)$$

$$S(U_{j+1}^{O_n}, L_l^{F_f}) \ge C(U_j^{O_n}, L_l^{F_f}) + M_s \times (1 - X_{(U_j^{O_n}, L_l^{F_f})}), \forall s \in S, j,n,l,f \quad (23)$$

$$\sum_{j=1}^{J} \sum_{n}^{N} \sum_{l}^{L} \sum_{f}^{F} X_{(U_j^{O_n}, L_l^{F_f})} = 1, \forall j,n,l,f \quad (24)$$

Eq. 15 defines the two objectives: 1) maximising factory loading balance and 2) minimising the order total tardiness and earliness. Eq. 16 defines that the latest task completion across all production lines determines the maximum completion time. Eqs. 17 and 18 enforce non-negativity constraints on task start and completion time. Eq. 19 guarantees completion time always over the starting time. Eqs. 20 and 21 ensure that consecutive tasks on the same production line do not overlap. Eq. 22 enforces non-preemptive processing, meaning each task is processed without interruption. Furthermore, Eq. 23 ensures that a successor task does not start before the predecessor completes. Finally, Eq. 24 guarantees that each task is assigned to one factory and one production line.

4. Solution Procedure for the LSTM-DHHFSP Model

4.1 Prediction of PCT matrix using LSTM

The previous section introduced an enhanced LSTM network to predict the PCT. Specifically, the cycle time of a given production line is represented as shown in Eq. 25.

$$T(L_l^{F_f}) = [T(L_{l,M_1}^{F_f}), T(L_{l,B_1}^{F_f}), \ldots, T(L_{l,B_b}^{F_f}), T(L_{l,M_s}^{F_f})] \quad (25)$$

Since different product types exhibit varying cycle times, the cycle time for a specific product type k is given by Eq. 26.

$$T(L_l^{F_f},k)=[T(L_{l,M_1}^{F_f},k),T(L_{l,B_1}^{F_f},k),...,T(L_{l,B_b}^{F_f},k),T(L_{l,M_s}^{F_f},k)] \quad (26)$$

Moreover, heterogeneous production line layouts lead to variations in cycle times. The PCT for different product types across all production stages can be expressed as the following matrix in Eq. 27.

$$\begin{bmatrix} T(L_{1,M_1}^{F_f},k),T(L_{1,B_1}^{F_f},k),\ldots,T(L_{1,B_b}^{F_f},k),T(L_{1,M_s}^{F_f},k) \\ T(L_{2,M_1}^{F_f},k),T(L_{2,B_1}^{F_f},k),\ldots,T(L_{2,B_b}^{F_f},k),T(L_{2,M_s}^{F_f},k) \\ \cdots \\ T(L_{l,M_1}^{F_f},k),T(L_{l,B_1}^{F_f},k),\ldots,T(L_{l,B_b}^{F_f},k),T(L_{l,M_s}^{F_f},k) \end{bmatrix} \quad (27)$$

4.2 Improved multi-objectives evolutionary algorithms for solving DHHFSP

In tackling hybrid flow-shop scheduling problems, MOEAs have emerged as powerful approaches due to their ability to handle multiple conflicting objectives.[5] Unlike traditional optimisation techniques that often struggle to balance trade-offs among competing goals, MOEAs are explicitly designed to approximate the Pareto-optimal front by guiding the search towards convergence while simultaneously preserving solution diversity. By leveraging evolutionary principles, they explore complex solution spaces effectively, ensuring both high-quality solutions and a well-distributed set of trade-offs across objectives.[26]

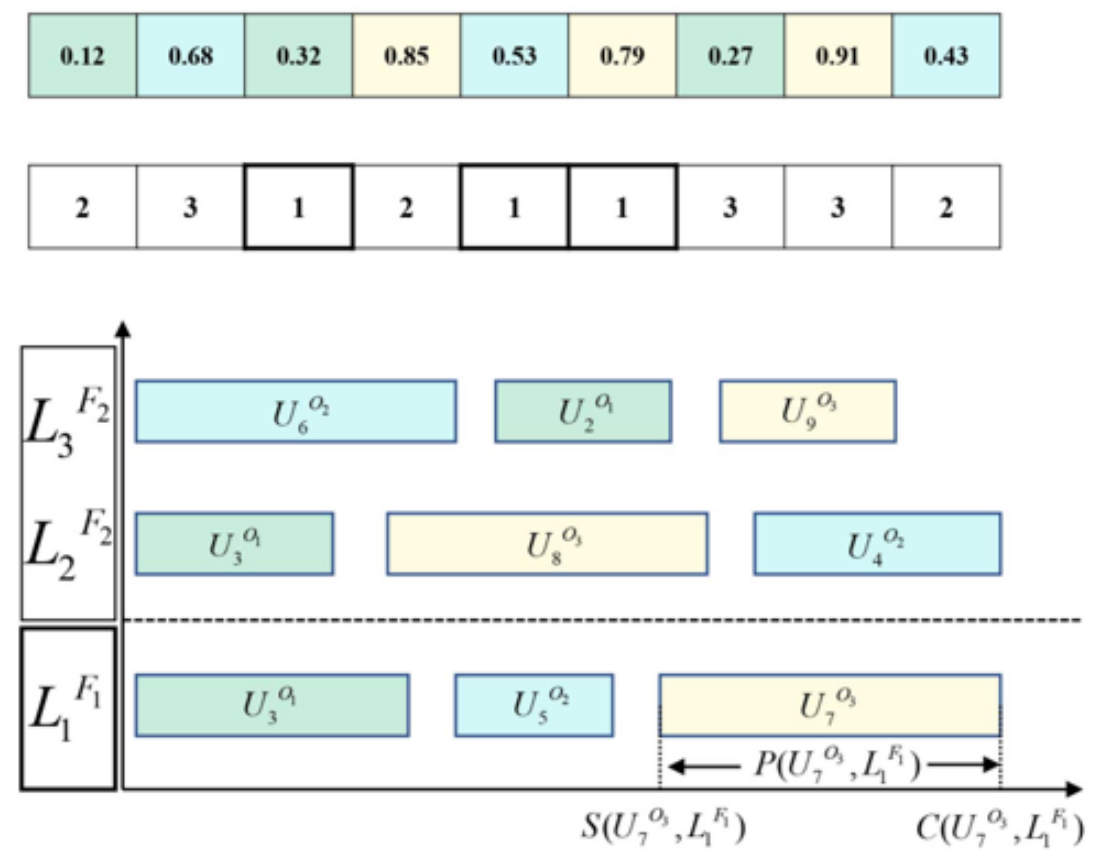


Fig. 2: Solution representation with encoding and decoding Gantt chart.

4.2.1 Encoding and decoding strategies

In the proposed algorithms, each candidate solution is represented by a double-vector encoding scheme, as illustrated in Fig. 2. This encoding mechanism consists of two distinct vectors: the first vector, termed as the *task sequence vector*, has a length equal to the total number of tasks, $Q(U_j^{O_n})$. Each element in this vector is a randomly generated number within the interval [0, 1], the lower value indicates a higher priority, thus dictating the processing sequence of tasks.

The second vector, the *production line assignment vector*, contains elements corresponding to a specific production line identifier, $L_l^{F_f}$. Each value in this vector determines the production line to which a given task is allocated. Together, these two vectors comprehensively represent task sequencing and production line assignment, enabling the algorithms to explore and evaluate diverse scheduling configurations.

As shown in Fig. 2, to further illustrate this solution representation, consider an example comprising three production orders: $O_1=\{U_1^{O_1},U_2^{O_1},U_3^{O_1}\}$, $O_2=\{U_4^{O_2},U_5^{O_2},U_6^{O_2}\}$, and $O_3=\{U_7^{O_3},U_8^{O_3},U_9^{O_3}\}$, comprising a total of nine tasks that need be assigned across two factories and three production lines. The production lines are distributed as follows: Factory 1 consists of a single production line, $F_1=\{L_1^{F_1}\}$, while Factory 2 includes two production lines, $F_2=\{L_2^{F_2},L_3^{F_2}\}$.

4.2.2 Solution initialisation and iteration optimisation

Designing a problem feature-based initialisation strategy can significantly enhance convergence speed and overall solution quality.[17] This study aims to balance the workload across multiple production lines while minimising variation in task completion times within the same order. Heuristic initialisation is employed to generate high-quality initial solutions, supplemented to maintain population diversity. This method prevents premature convergence and facilitates a more comprehensive solution for space exploration.

Algorithm 1: Heuristic initialisation strategy.

Require: Set of orders $\{O_1, O_2, \cdots, O_n\}$. Set of tasks $\left\{U_1^{O_n}, U_2^{O_n}, \cdots, U_j^{O_n}\right\}$. Set of production lines $\left\{L_1^{F_f}, L_2^{F_f}, \cdots, L_l^{F_f}\right\}$. Processing time matrix. *PCT* Population size *PS*.

Initialise population counter $PS_i \leftarrow 0$
while $PS_i < PS/2$ do ▷ First half of population generation
for each order O_n, where $n \leq N$ do
Randomly select an order O_n and its associated tasks
for each task $U_j^{O_n}$, where $j \leq J$ do
Compare the processing time $P(U_j^{O_n}, L_l^{F_f})$ for task j of order O_n for each production line l of factory F_f
Assign the task to the first available production line with the lowest processing time
end for
end for
Increment population counter: $PS_i \leftarrow PS_i + 1$
end while
while $PS/2 < PS_i < PS$ do ▷ Second half of population generation
Randomly generate new solutions
Increment population counter: $PS_i \leftarrow PS_i + 1$
end while
return Initialised task schedule

Algorithm 1 *Heuristic initialisation strategy* outlines a heuristic approach for solution initialisation. To ensure population diversity, half of the solutions are generated using a heuristic generate strategy, while the remainder are generated randomly.

During the iterative process, MOEAs evaluate the fitness of individual solutions, employing a non-dominated selection mechanism to guide evolutionary progress. Furthermore, at each iteration, a local search strategy integrates Simulated Binary Crossover (SBX) and polynomial mutation to enhance the exploration of promising regions within the search space.[25] To prevent indefinite execution, an upper bound on the number of iterations is imposed, and the algorithm terminates upon reaching this predefined threshold.

5. Experimentation analysis

The model was developed and validated using Python 3.7 and executed on a PC with a Windows 10 operating system, an Intel Core i5 processor (3.50 GHz), and 32GB of RAM. The proposed PCT prediction model, implemented using the LSTM network, was tested and validated with historical data collected from the production floor. Subsequently, the PCT matrix generated by the LSTM model was applied in conjunction with MOEAs to guide the scheduling of the DHHFSP, providing a structured framework for optimising.

5.1 LSTM-based prediction of PCT

5.1.1 System parameters and training data

The production line in this case study comprises three workstations corresponding to the stages of panel furniture processing: cutting, edge-banding, and drilling. Additionally, two buffers are positioned between the workstations to support

workflow. The dataset, collected from daily operations on the plant floor, includes data from over 10,000 products spanning ten distinct product types. The dataset was randomly split into 70% for training and 30% for testing.

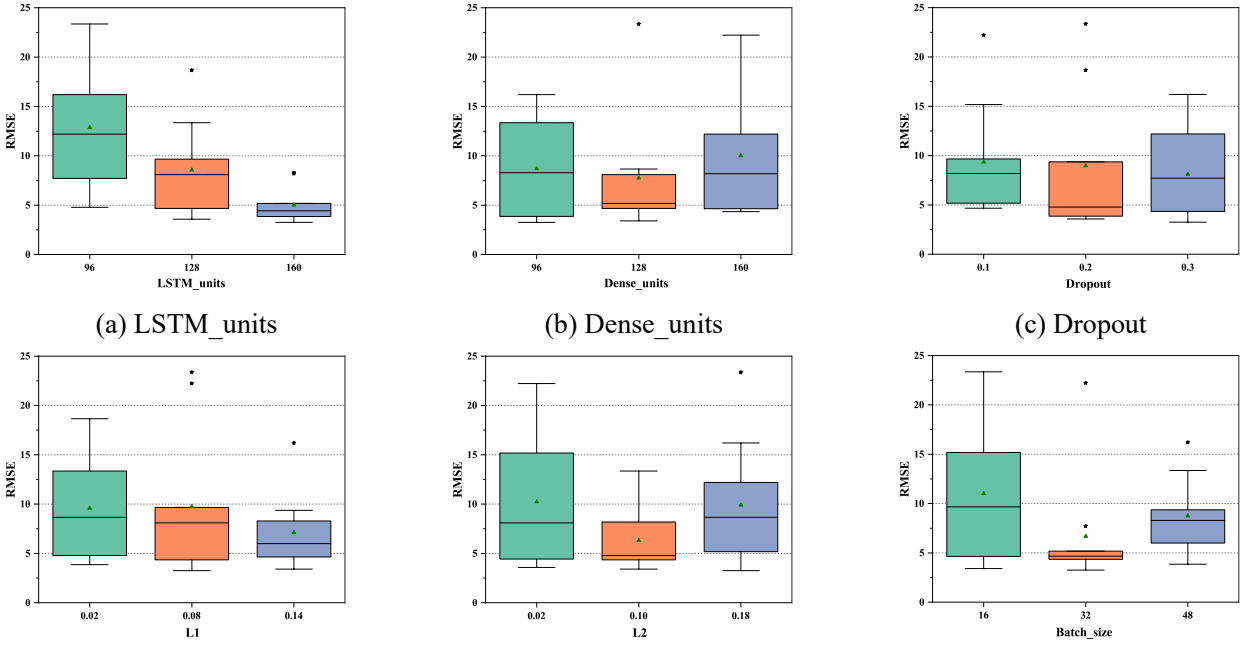
The optimiser used in this study is RMSprop, with a learning rate $\alpha = 0.0001$ and a maximum of 400 epochs, while all other parameters are set to their default values. The LSTM network was implemented using the TensorFlow platform (version 2.1.0). To determine the most suitable configuration, a series of experiments were conducted by varying key hyper-parameters, including the number of LSTM units, dense units, dropout rate, L1/L2 regularisation, and batch size. The candidate settings for these parameters are summarised in Table 1.

Table 1: Candidate hyper-parameter levels of the LSTM network.

Hyper-parameters	Level-1	Level-2	Level-3
LSTM_units	96	128	160
Dense_units (FC Layer)	96	128	160
Dropout	0.10	0.20	0.30
L1 regularisation	0.02	0.08	0.14
L2 regularisation	0.02	0.10	0.18
Batch_size	16	32	48

The selection of these hyper-parameters is based on both theoretical considerations and empirical validation. Specifically, the number of LSTM units (96, 128, 160) was chosen to balance model expressiveness and computational efficiency, ensuring sufficient capacity to capture complex temporal dependencies in production data without incurring excessive training costs. Dropout rates (0.1–0.3) were introduced to mitigate overfitting, which is particularly important given the variability and uncertainty of shop-floor data. Regularisation terms (L1: 0.02–0.14; L2: 0.02–0.18) were included to further constrain the model complexity and improve generalisation ability. The batch sizes (16, 32, 48) were tested to stabilise gradient updates, with medium-scale batches providing the best trade-off between training efficiency and predictive accuracy.

Root Mean Squared Error (RMSE), which measures the average magnitude of prediction errors, was used to evaluate the accuracy of the LSTM network under different hyper-parameter settings. As shown in Fig. 3, the number of LSTM units had a clear impact on performance: increasing the units from 96 to 160 consistently reduced RMSE and yielded more stable predictions. Dense units and Dropout provided



(a) LSTM_units (b) Dense_units (c) Dropout

(d) L1 (e) L2 (f) Batch_size

Fig. 3: Experimental results of LSTM hyper-parameter tuning. Boxplots show the effect of (a) LSTM units, (b) Dense units, (c) Dropout, (d) L1 regularisation, (e) L2 regularisation, and (f) Batch size on RMSE.

regularisation benefits at moderate levels. Similarly, both L1 and L2 regularisation improved generalisation when set to medium values (0.08 and 0.10, respectively), whereas overly small or large values caused instability. For batch size, a medium value of 32 achieved the lowest RMSE and tightest distribution, while very small (16) or large (48) batches resulted in higher errors. Overall, these results indicate that the configuration with 160 LSTM units, 128 Dense units, dropout = 0.2, L1 = 0.08, L2 = 0.10, and batch size = 32 provides the best trade-off between predictive accuracy and robustness.

5.1.2 Model training and performance evaluation

To examine the impact of dataset size on the performance of the LSTM model, the input data is divided into three groups containing 1,000, 5,000, and 10,000 samples, respectively. The model's performance is evaluated using five standard

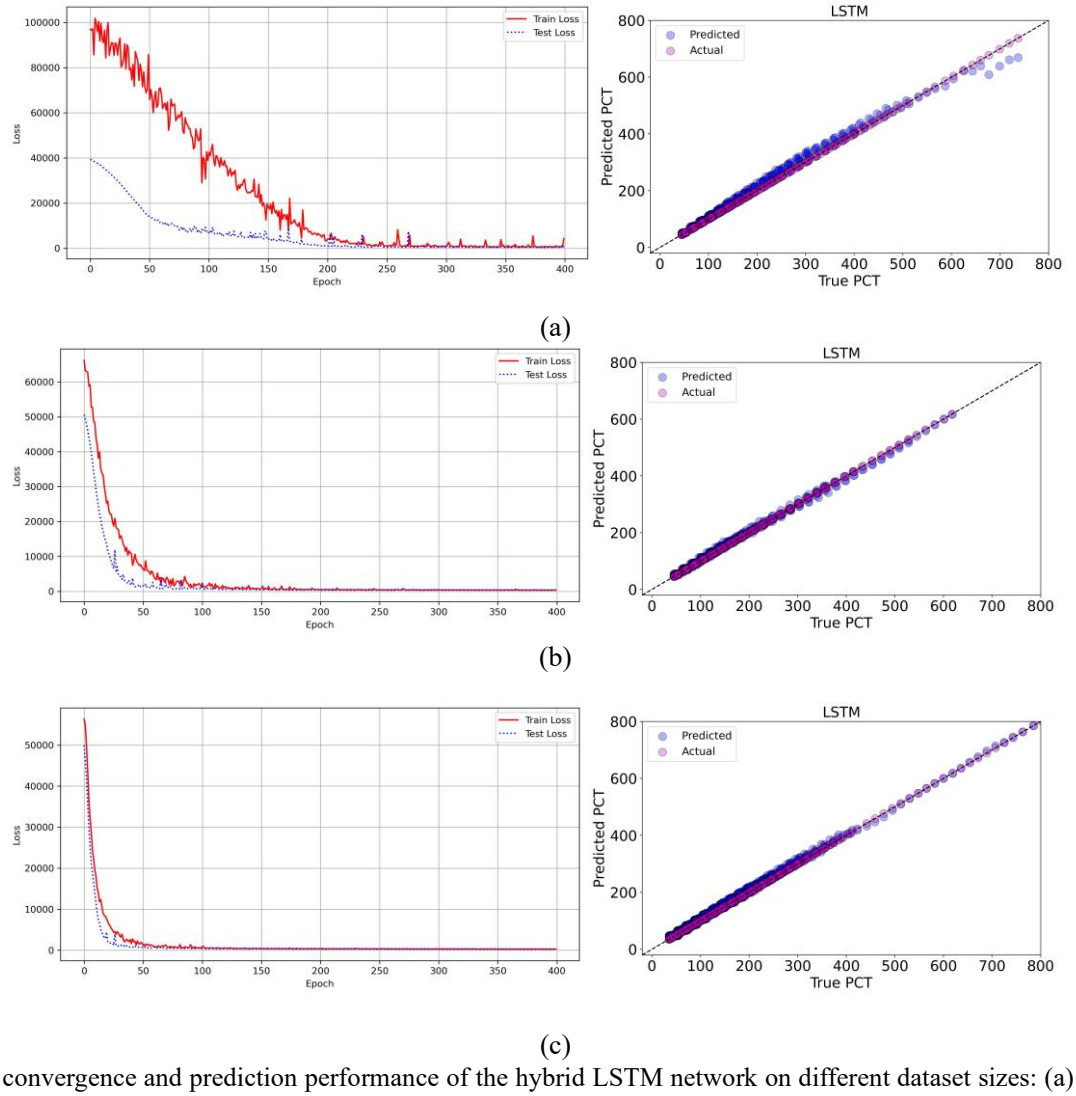


Fig. 4: Training convergence and prediction performance of the hybrid LSTM network on different dataset sizes: (a) 1,000 dataset, (b) 5,000 dataset and (c) 10,000 dataset.

regression metrics: Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE), Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and coefficient of determination (R^2). Each dataset configuration is run ten times, and the average results are recorded to ensure statistical reliability.

The training process and prediction performance is depicted in Fig. 4, which illustrates the progressive reduction in both training and testing losses as the number of epochs increases. When trained on 1,000 samples, the model requires approximately 200 epochs for the loss to stabilise. With 5,000 samples, convergence occurs more rapidly, requiring around 100 epochs, whereas training on 10,000 samples led to even faster stabilisation within 50 epochs, with both training and testing losses reaching consistently low values.

As illustrated in Fig. 4, the prediction accuracy of the LSTM model improves as the dataset size increases. With 1,000 samples (Fig. 4a), noticeable deviations from the diagonal line (y = x) can be observed, indicating unstable generalisation. When the dataset size expands to 5,000 samples (Fig. 4b), the dispersion decreases substantially, and predictions align more closely with actual values. The best performance is achieved with 10,000 samples (Fig. 4c), where the predicted values almost perfectly coincide with the diagonal, reflecting minimal error and excellent robustness.

The evaluation results, summarised in Table 2, reveal a clear trend of enhanced model performance with increasing dataset size. When trained on 1,000 samples, the model exhibits relatively high errors (MAE: 7.300, MAPE: 5.05%) and lower predictive accuracy (R^2: 0.909). However, expanding to 5,000 samples significantly reduces the errors (MAE: 4.804, MAPE: 2.87%) and improves the R^2 value to 0.927. The best performance is achieved with 10,000 samples,

Table 2: Performance comparison among different dataset size.

Data size	1,000	5,000	10,000
MAE	7.300	4.804	**2.176**
MAPE	5.05%	2.87%	**1.32%**
MSE	102.54	55.316	**15.379**
RMSE	9.851	7.241	**3.559**
R^2	0.909	0.927	**0.959**

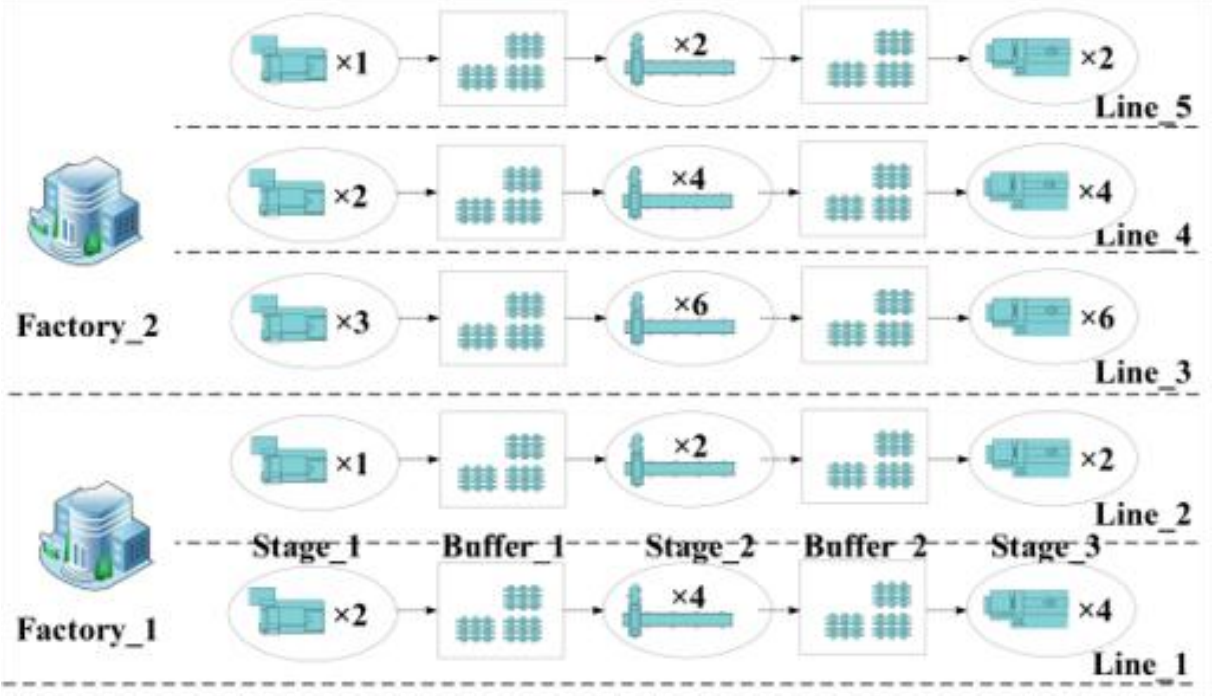


Fig. 5: Production lines layout of furniture manufacturing factories.

where the MAE decreases to 2.176, the MAPE drops to 1.32%, and the R^2 value increases to 0.959, indicating a stronger correlation between the predicted and actual values.

5.2 Case study: Panel furniture DHHFSP with LSTM-predicted PCT

5.2.1 Manufacturing environment setup

As illustrated in Fig. 5, the manufacturing environment in this case study consists of two distributed factories, incorporating a total of five production lines: two in Factory 1 and three in Factory 2. Each production line comprises three processing stages, with two buffers between consecutive workstations to facilitate workflow. This scenario is classified as DHHFSP due to the variability in cycle times across production lines. Details of cycle time information are provided in Table 3.

The order data is derived from Section 5.1. In this case

Table 3: Cycle time (in seconds) for each stage of the production lines.

		Stage 1	Buffer 1	Stage 2	Buffer 2	Stage 3
F_1	L_1	(15, 20)	(5, 10)	(6, 10)	(5, 10)	(10, 20)
	L_2	(8, 10)	(5, 10)	(3, 5)	(5, 10)	(5, 10)
F_2	L_1	(15, 20)	(5, 10)	(6, 10)	(5, 10)	(10, 20)
	L_2	(8, 10)	(5, 10)	(3, 5)	(5, 10)	(5, 10)
	L_3	(5, 7)	(5, 10)	(2, 4)	(5, 10)	(5, 7)

Table 4: Partial PCT matrix of tasks in different production lines.

Order No.	Task No.	Product type	PCT in F1(L1)	PCT in F1(L2)	PCT in F2(L1)	PCT in F2(L2)	PCT in F2(L3)
1	1	B	689	389	725	386	272
1	2	C	204	109	182	106	65
1	3	D	185	109	193	115	65
2	4	C	149	89	155	86	62
2	5	D	185	109	193	115	65
3	6	E	126	69	110	71	48
23	73	A	113	73	118	66	48
23	74	B	167	99	174	96	69
23	75	C	59	39	60	38	23
23	76	E	126	69	110	71	48

study, 23 orders encompass 76 tasks, involving approximately 900 products across five distinct product types A, B, C, D, and E. The PCT for these tasks across multi-production lines has been pre-processed using the LSTM model. Table 4 provides a partial PCT matrix.

5.2.2 Comparison experiments and verification

In this subsection, the efficacy and performance of four MOEAs and their heuristic-enhanced variants are evaluated for solving the DHHFSP. Parameter sensitivity experiments were first carried out to determine the optimal configuration for each algorithm. Based on these optimal settings, the overall performance of the four MOEAs was then compared. The contribution of heuristic initialisation was further examined by contrasting the standard versions with their heuristic-enhanced counterparts. Finally, the scheduling results were visualised and analysed using Gantt charts to provide an intuitive illustration of the optimisation outcomes.

Parameter sensitivity experiments were conducted to investigate the effect of different parameter choices on the performance of the MOEAs. The tested parameters and their candidate levels are summarised in Table 5. To ensure a fair comparison across algorithms, the population size and the number of iterations were uniformly set to 100. Other algorithmic settings followed the standard implementations of each MOEA.

The results of the parameter sensitivity experiments are shown in Fig. 6. Mean hypervolume (HV) is adopted as the performance indicator to evaluate the quality of the Pareto front obtained under different parameter settings. HV measures the volume in the objective space dominated by the solution set, with larger values indicating better convergence and diversity. To reduce randomness, the results are averaged over multiple runs, and the mean HV is reported.

Table 5: Candidate parameter settings of various MOEAs.

Algorithms	Symbol	Parameters	Meaning	Levels
NSGA-II[26]	pc	Crossover probability	Probability of applying crossover operator between two individuals	0.4, 0.6, 0.8
	pm1	Mutation probability 1	Probability of mutation at gene/bit level, enhancing solution diversity	0.70, 0.85, 0.95
	pm2	Mutation probability 2	Probability of parameter-level mutation, further maintaining diversity	0.03, 0.05, 0.08
MOPSO[27]	c1	Cognitive coefficient	Weight for individual learning, guiding particles toward their own best position	1.2, 1.6, 2.0
	c2	Social coefficient	Weight for social learning, guiding particles toward the global best	1.2, 1.6, 2.0
	Vmax	Maximum velocity	Upper bound of particle velocity, balancing global exploration and local exploitation	0.10, 0.15, 0.20
MODEA[28]	F	Differential weight	Scaling factor of differential variation, controlling mutation strength	0.4, 0.6, 0.8
	CR	Crossover rate	Probability of mixing parent vector with differential vector	0.70, 0.85, 0.95
	pm2	Mutation probability 2	Additional parameter mutation probability, increasing diversity	0.03, 0.05, 0.08
SPEA2[29]	pc	Crossover probability	Probability of applying crossover operator between two individuals	0.80, 0.90, 0.95
	pm1	Mutation probability 1	Probability of mutation at gene/bit level, enhancing solution diversity	0.04, 0.08, 0.12
	pm2	Mutation probability 2	Probability of parameter-level mutation, further maintaining diversity	0.03, 0.05, 0.07

As illustrated in Fig. 6, the performance of each MOEA is highly sensitive to parameter choices. For NSGA-II (Fig. 6(a)), a lower crossover probability (pc = 0.80) yields the best HV, mutation probabilities pm1 = 0.04 and pm2 = 0.03 provide a good balance between diversity and stability. For MOPSO (Fig. 6(b)), the cognitive coefficient c1 = 1.2 achieves significantly better performance than higher values, while c2 = 2.0 enhances exploration. A smaller velocity bound (Vmax = 0.10) also improves convergence. For MODEA (Fig. 6(c)), the algorithm benefits from stronger differential variation (F = 0.8) and crossover rate (CR = 0.95), and a moderate mutation probability pm2 = 0.05, which together deliver the highest HV. For SPEA2 (Fig. 6(d)), performance is maximised with pc = 0.90, pm1 = 0.04, and pm2 = 0.03, showing that carefully tuned mutation rates are crucial for maintaining population diversity.

Overall, the results confirm that appropriate parameter tuning has a substantial impact on algorithmic performance. Based on the sensitivity analysis shown in Fig. 6, the most effective parameter values for each algorithm were identified

and then applied in the following comparison experiments.

In the next step, the four MOEAs and heuristic initialisation-enhanced version are evaluated under their respective optimal parameter settings to provide a fair comparison of their overall performance in solving the DHHFSP. Three primary metrics are employed to evaluate the quality of solutions: Mean Ideal Distance (MID), Spread of Non-Dominated Solution (SNS) and Rate of Achievement to

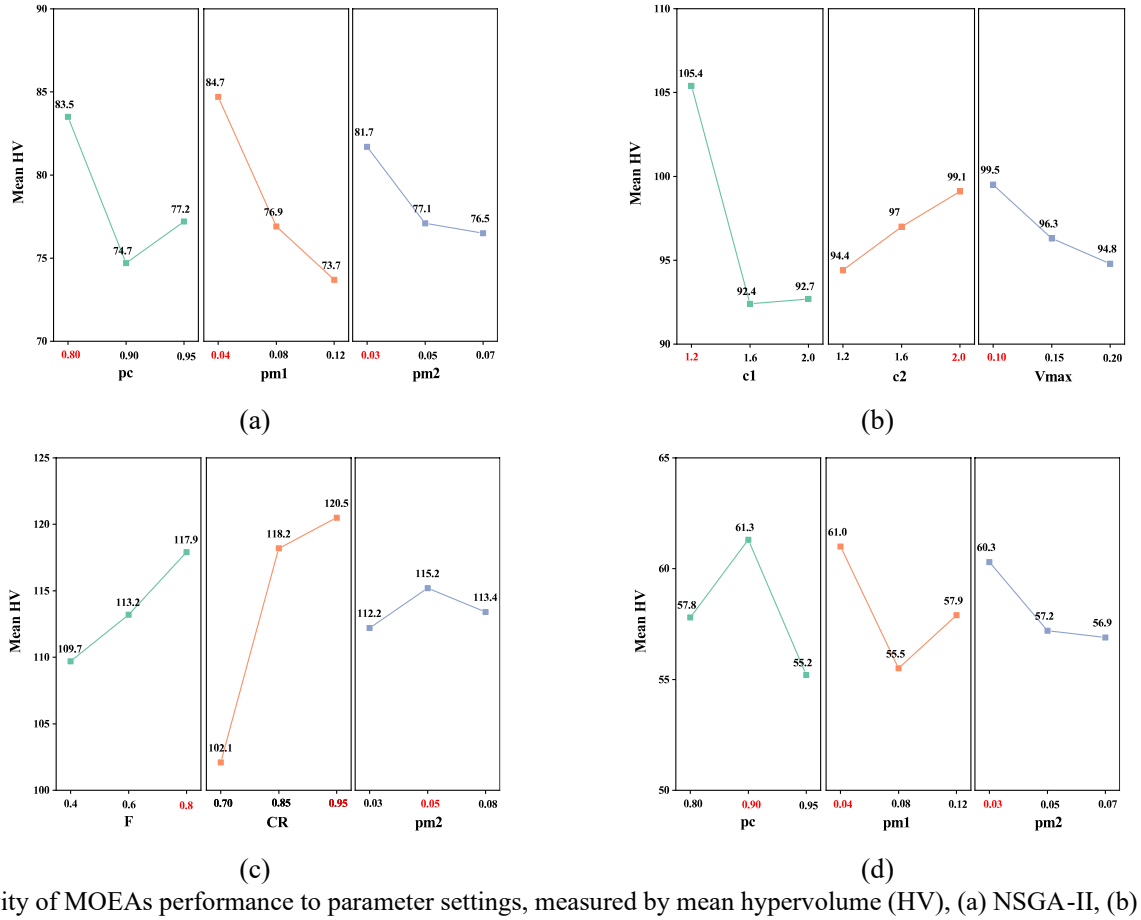


Fig. 6: Sensitivity of MOEAs performance to parameter settings, measured by mean hypervolume (HV), (a) NSGA-II, (b) MOPSO, (c) MODEA and (d) SPEA2.

Two Objectives simultaneously (RAS). Their definitions and interpretations are provided below. Note that the lower the value of MID and RAS, the higher the value of SNS, the better the solution quality.[30] In addition to the three primary performance metrics, the running time (RT) and the ideal value for two objectives, namely, Mean value of two objectives (Z_1) and (Z_2) are taken into consideration. Each experimental run is repeated ten times to ensure the reliability of the results, and the average values are recorded.

MID (Mean ideal distance): Measure the closeness between the Pareto solution and an ideal point, set as (1, 0). It is defined as in Eq. 28.

$$MID = \frac{\sum_{i=1}^{N} d_i}{N}, \tag{28}$$

where N is the number of non-dominated solutions and $di = \sqrt{\sum_{j=1}^{m}(f_j(x_i - z_j{}^*))^2}$, is the Euclidean distance between the i-th solution's objective value and the ideal point.

SNS: Measures the diversity of solutions within the Pareto front. It's calculated as in Eq. 29.

$$SNS = \frac{\sum_{i=1}^{N-1} \left| d_i - \bar{d} \right|}{(N-1)}, \tag{29}$$

where d_i is the Euclidean distance between adjacent solutions, $d_i = \|f(x_i) - f(x_i + 1)\|$.

RAS: Evaluates how well the obtained solutions balance two conflicting objectives. It is defined as in Eq. 30.

$$RAS = \frac{\sum_{i=1}^{N}(\frac{f_{1i} - F_i}{F_i}) + (\frac{f_{2i} - F_i}{F_i})}{N}, \tag{30}$$

where $F_i = min\{f_{1i}, f_{2i}\}$.

The comparative performance of the four MOEAs and their heuristic-enhanced variants is summarised in Table 6, while the visual comparisons in Fig. 7 further corroborate these findings. Among the standard algorithms, NSGA-II exhibits the strongest overall performance, achieving the lowest MID and the highest SNS, reflecting both closer convergence to the Pareto front and greater diversity of solutions. In addition, NSGA-II demonstrates competitive computational efficiency. As illustrated in Fig. 7(a), the radar chart confirms that NSGA-II outperforms the other algorithms across nearly all metrics under random initialisation.

Table 6: Comparative results of performance metrics for standard and heuristic-enhanced MOEAs.

	NSGA-II		MOPSO		MODEA		SPEA2	
	Ave.	Std.	Ave.	Std.	Ave.	Std.	Ave.	Std.
MID	**121.009**	53.435	134.758	67.816	147.966	33.472	150.007	79.618
SNS	**0.882**	0.059	0.819	0.082	0.826	0.080	0.839	0.083
RAS	**0.638**	0.076	0.663	0.097	0.711	0.115	0.675	0.061
RT	**14.487**	0.169	15.186	0.075	15.031	0.153	14.865	0.242
Z_1	0.970	0.014	**0.988**	0.012	0.953	0.008	0.969	0.009
Z_2	**896.673**	60.989	1141.129	68.840	1062.286	33.851	928.515	51.555
	h-NSGA-II		h-MOPSO		h-MODEA		h-SPEA2	
	Ave.	Std.	Ave.	Std.	Ave.	Std.	Ave.	Std.
MID	**80.906**	30.908	102.043	41.070	296.325	72.984	93.927	61.646
SNS	**0.869**	0.093	0.831	0.132	0.800	0.102	0.794	0.131
RAS	0.657	0.082	**0.647**	0.075	0.664	0.133	0.708	0.107
RT	**14.567**	0.115	15.628	0.398	15.730	0.212	15.029	0.125
Z_1	0.972	0.009	0.956	0.014	0.959	0.011	**0.978**	0.010
Z_2	480.137	50.612	639.516	86.499	905.466	60.184	**471.784**	63.126

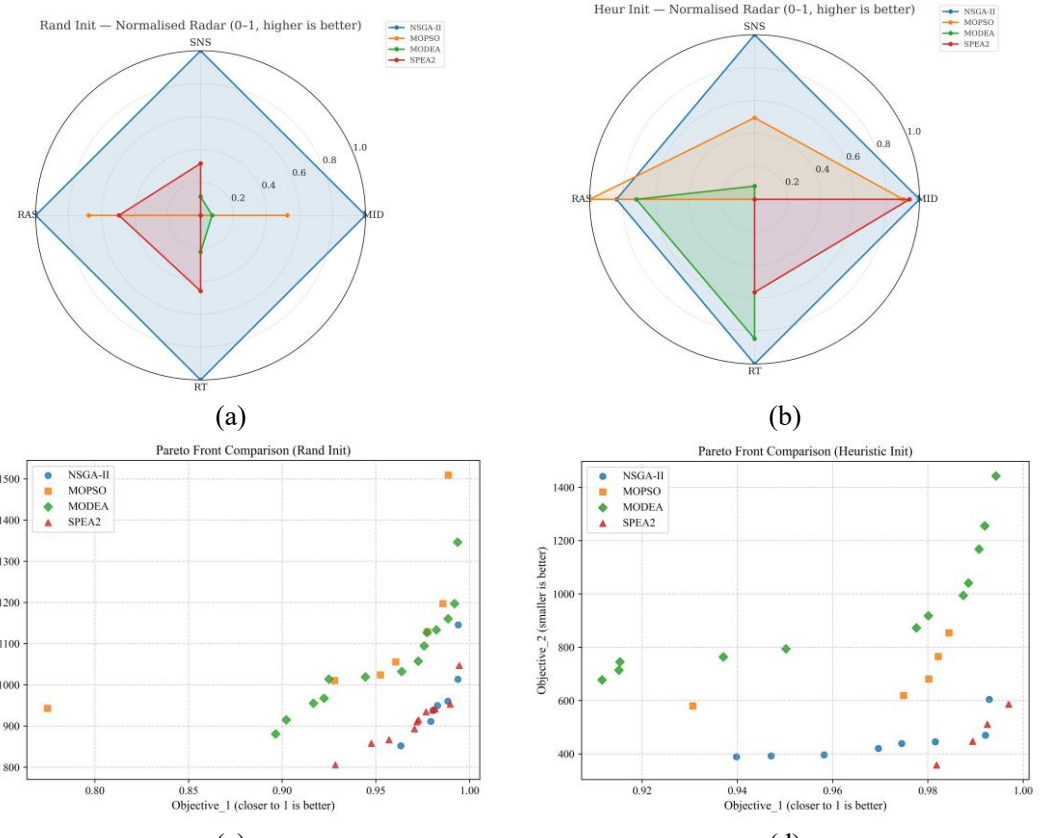
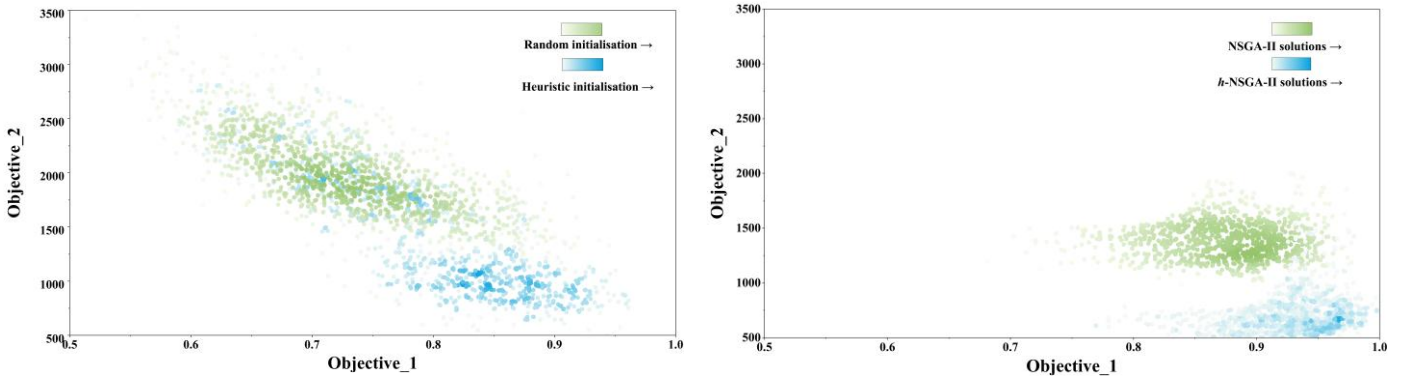


Fig. 7: Comparison of MOEAs with random and heuristic initialisation strategies using normalised radar evaluations (a–b) and Pareto fronts (c–d).

When heuristic initialisation is introduced, all algorithms show substantial performance gains. In particular, h-NSGA-II achieves further reductions in MID and notable improvements in SNS compared with its standard counterpart, highlighting the value of heuristic initialisation in generating high-quality and diverse solutions. As shown in Fig. 7(b), h-NSGA-II dominates across three of the four radar dimensions, establishing its superiority under heuristic-enhanced conditions.

The Pareto front distributions in Fig. 7(c)–(d) further reinforce these observations. Under random initialisation, NSGA-II and SPEA2 are the most competitive, producing well-converged and diverse fronts. However, with heuristic initialisation, solutions shift significantly closer to the ideal



(a) (b)

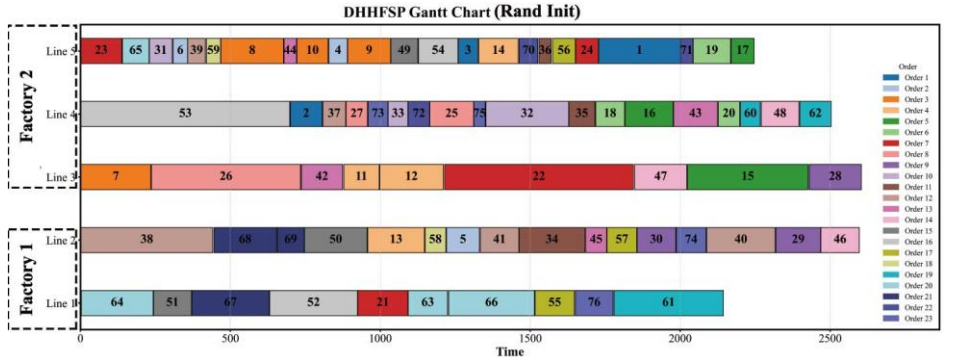
Fig. 8: Comparison of initial and optimised solutions relative to the ideal point (1, 0): (a) random vs. heuristic initialisation, (b) standard NSGA-II vs. h-NSGA-II.

front. Although h-SPEA2 achieves the most extreme Pareto values, its lack of diversity reduces its overall effectiveness. In contrast, h-NSGA-II provides a more balanced front, combining strong convergence with well-preserved diversity.

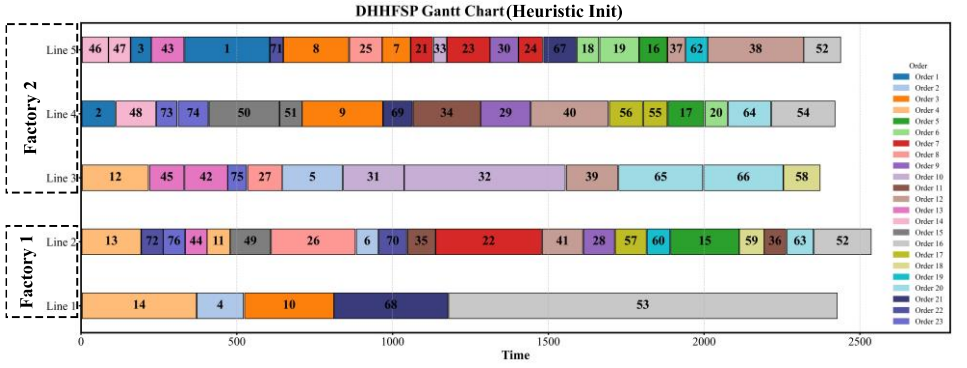
Overall, h-NSGA-II emerges as the most effective algorithm, delivering the best trade-off between convergence, diversity, and computational efficiency. These results confirm that the proposed heuristic initialisation mechanism substantially enhances MOEA performance, making it particularly effective for solving the DHHFSP under uncertain processing times.

To further illustrate the advantages of heuristic initialisation, Fig. 8 presents density accumulation maps for both initial and optimised solutions. As shown in Fig. 8(a), random initialisation yields solutions clustered around Objective 1 ≈ 0.75 and Objective 2 ≈ 2000, whereas heuristic initialisation produces a more favourable distribution around Objective1 ≈ 0.85 and Objective 2 ≈ 1000. Fig. 8(b) highlights the final solutions of NSGA-II and h-NSGA-II, where the hybrid variant converges much closer to the ideal point (1, 0). This reflects not only improved convergence behaviour but also a more balanced workload distribution and reduced earliness/tardiness across orders.

Fig. 9 reports the scheduling results of the DHHFSP using



(a)



(b)

Fig. 9: Comparison of scheduling schemes for the DHHFSP under random initialisation (a) and heuristic initialisation (b) using Gantt charts.

NSGA-II under its tuned optimal settings. Tasks are labeled by index; operations belonging to the same order share identical colours.

In Fig. 9(a) (random initialisation), several long contiguous processing blocks and idle intervals can be observed on individual lines, indicating uneven machine utilisation across factories and lines. The completion times of operations within the same order are relatively dispersed, implying larger intra-order earliness/tardiness and a longer tail near the makespan.

In contrast, Fig. 9(b) (heuristic initialisation) exhibits a visibly more even allocation of workload across Factory 1 and Factory 2 with fewer large gaps, shorter late tails, and denser operations. Operations from the same order tend to complete within a narrower time window, reflecting reduced intra-order earliness/tardiness and improved flow synchronisation. The overall horizon is also shortened, suggesting a lower makespan and higher line utilisation.

These visual observations are consistent with the quantitative results reported earlier: the heuristic-enhanced setting yields better convergence to the ideal point and improved diversity (lower MID/RAS, higher SNS). Collectively, the Gantt charts confirm that heuristic initialisation, combined with optimally tuned NSGA-II, produces schedules with more balanced cross-factory loads and smaller order-level deviations.

6. Conclusion

This paper addresses the Distributed Heterogeneous Hybrid Flow-Shop Scheduling Problem (DHHFSP) under uncertain Production Completion Times (PCT), proposing an innovative solution that integrates Deep Learning and Multi-Objective Evolutionary Algorithms (MOEAs). Specifically, an enhanced Long Short-Term Memory (LSTM) network is developed to capture temporal dependencies in processing times. By modelling product types and machine capabilities as recurrent units, the model accurately predicts PCT across production lines with varying capacities.

Extensive validation across different data sizes indicates that the best performance is achieved with 10,000 samples, yielding a Mean Absolute Error (MAE) of 2.176, a Mean Absolute Percentage Error (MAPE) of 1.32%, and an R^2 value of 0.959. Training and test loss are significantly reduced as the dataset size increases, leading to precise PCT predictions.

The predicted processing time matrix is then subsequently integrated into an MOEA to solve the DHHFSP, with the objective of minimising order delays and balancing the factory workload. A novel constructive heuristic algorithm is introduced to generate high-quality initial solutions, further enhancing the MOEA's optimisation capabilities.

A comparison analysis of four standard and heuristic-enhanced MOEAs (NSGA-II, MOPSO, MODEA, and SPEA2) using a real-world case study demonstrates the effectiveness of the proposed approach. The heuristic-enhanced NSGA-II (h-NSGA-II) achieves the best overall performance, yielding the lowest Mean Ideal distance (MID), the highest solution diversity within the Pareto front (SNS), and the most balanced optimisation of the two objectives.

The integration of LSTM-based prediction with evolutionary optimisation provides distinct advantages in environments characterised by uncertainty. Unlike conventional deterministic models, which assume fixed or uniformly distributed processing times and thus fail to reflect real manufacturing variability, the proposed framework learns the nonlinear and stochastic nature of processing times and directly incorporates this knowledge into scheduling decisions. This not only enhances robustness against disturbances but also improves responsiveness and decision quality in distributed production systems. The findings highlight the practical potential of the combined LSTM–MOEA approach as a robust and intelligent scheduling strategy for complex, uncertain manufacturing environments.

Acknowledgments

This work is supported by National Natural Science Foundation of China [32471791]. The project from Technology Innovation Alliance of Wood/Bamboo Industry; the International Cooperation Joint Laboratory for Production, Education, Research and Application of Ecological Health Care on Home Furnishing. Specially acknowledge to ZBOM Home Collection CO.,LTD for provided production data.

Conflict of Interest

The authors report no declarations of interest.

Data statement

The Python code and data about this paper can be found through this link: https://github.com/Sherwin-WU/LSTM-DHHFSP.git

Supporting Information

Not applicable.

CRediT Statement

Zhanwen Wu: Original draft, Review & editing, Validation, Software, Methodology. **Weihang Dong:** Software, Methodology. **Jinxin Wang:** Review & editing. **Feng Zhang** and **Zhaolong Zhu:** Investigation, Validation. **Xiaolei Guo:** Resources, Funding acquisition. **Pingxiang Cao:** Supervision.

References

[1] Y. Li, F. Li, Q.-K. Pan, L. Gao, M. F. Tasgetiren, An artificial bee colony algorithm for the distributed hybrid flowshop scheduling problem, *Procedia Manufacturing*, 2019, **39**, 1158-1166, doi: 10.1016/j.promfg.2020.01.354.

[2] J. W. Veile, D. Kiel, J. M. Müller, K.-I. Voigt, Lessons learned from Industry 4.0 implementation in the German manufacturing industry, *Journal of Manufacturing Technology Management*, 2019, **31**, 977-997, doi: 10.1108/jmtm-08-2018-0270.

[3] B.-K. Yang, R.-R. Li, Y. Meng, Z.-Y. Xu, Research on texture

features classification of multilayered wood flooring using global image structure method, *Wood Material Science & Engineering*, 2025, 1-13, doi: 10.1080/17480272.2025.2453625.
[4] X. Xiong, Q. Ma, YingyingYuan, Z. Wu, M. Zhang, Current situation and key manufacturing considerations of green furniture in China: a review, *Journal of Cleaner Production*, 2020, **267**, 121957, doi: 10.1016/j.jclepro.2020.121957.
[5] J. Wang, Z. Wu, L. Yang, W. Hu, C. Song, Z. Zhu, X. Guo, P. Cao, Investigation on distributed rescheduling with cutting tool maintenance based on NSGA-III in large-scale panel furniture intelligent manufacturing, *Journal of Manufacturing Processes*, 2024, **112**, 214-224, doi: 10.1016/j.jmapro.2024.01.016.
[6] X. Zhou, R. Li, Z. Wu, Scheduling optimization for laminated door machining shop based on improved genetic algorithm, *Computers & Operations Research*, 2025, **180**, 107078, doi: 10.1016/j.cor.2025.107078.
[7] W. Shao, Z. Shao, D. Pi, Modeling and multi-neighborhood iterated greedy algorithm for distributed hybrid flow shop scheduling problem, *Knowledge-Based Systems*, 2020, **194**, 105527, doi: 10.1016/j.knosys.2020.105527.
[8] L. Luo, X. Yan, Scheduling of stochastic distributed hybrid flow-shop by hybrid estimation of distribution algorithm and proximal policy optimization, *Expert Systems with Applications*, 2025, **271**, 126523, doi: 10.1016/j.eswa.2025.126523.
[9] I. Ansari, M. Barati, M. R. Sadeghi Moghadam, M. Ghobakhloo, Empowering Industry 4.0: systemizing readiness assessment and strategic roadmapping for digital transformation, *Journal of Manufacturing Technology Management*, 2025, **36**, 937-960, doi: 10.1108/jmtm-05-2024-0257.
[10] R. Buehler, J. Peetz, D. Griffin, Finishing on time: When do predictions influence completion times?, *Organizational Behavior and Human Decision Processes*, 2010, **111**, 23-32, doi: 10.1016/j.obhdp.2009.08.001.
[11] M. Yuan, Z. Li, C. Zhang, L. Zheng, K. Mao, F. Pei, Research on real-time prediction of completion time based on AE-CNN-LSTM, *Computers & Industrial Engineering*, 2023, **185**, 109677, doi: 10.1016/j.cie.2023.109677.
[12] J. Huang, Q. Chang, J. Arinez, Product completion time prediction using a hybrid approach combining deep learning and system model, *Journal of Manufacturing Systems*, 2020, **57**, 311-322, doi: 10.1016/j.jmsy.2020.10.006.
[13] H. Cui, X. Li, L. Gao, An improved multi-population genetic algorithm with a greedy job insertion inter-factory neighborhood structure for distributed heterogeneous hybrid flow shop scheduling problem, *Expert Systems with Applications*, 2023, **222**, 119805, doi: 10.1016/j.eswa.2023.119805.
[14] H. Öztop, M. Fatih Tasgetiren, D. T. Eliiyi, Q.-K. Pan, Metaheuristic algorithms for the hybrid flowshop scheduling problem, *Computers & Operations Research*, 2019, **111**, 177-196, doi: 10.1016/j.cor.2019.06.009
[15] I. Ribas, R. Leisten, J. M. Framiñan, Review and classification of hybrid flow shop scheduling problems from a production system and a solutions procedure perspective, *Computers & Operations Research*, 2010, **37**, 1439-1454, doi: 10.1016/j.cor.2009.11.001.
[16] J.-Q. Li, X.-L. Chen, P.-Y. Duan, J.-H. Mou, KMOEA: a knowledge-based multiobjective algorithm for distributed hybrid flow shop in a prefabricated system, *IEEE Transactions on Industrial Informatics*, 2022, **18**, 5318-5329, doi: 10.1109/TII.2021.3128405.
[17] Z. Shao, W. Shao, D. Pi, LS-HH: a learning-based selection hyper-heuristic for distributed heterogeneous hybrid blocking flow-shop scheduling, *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2023, **7**, 111-127, doi: 10.1109/TETCI.2022.3174915.
[18] F. Zhao, F. Yin, L. Wang, Y. Yu, A Co-Evolution Algorithm With Dueling Reinforcement Learning Mechanism for the Energy-Aware Distributed Heterogeneous Flexible Flow-Shop Scheduling Problem, *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2025, **55**(3), 1794-1809, doi: 10.1109/TSMC.2024.3510384.
[19] Z. Pan, D. Lei, L. Wang, A Knowledge-Based Two-Population Optimization Algorithm for Distributed Energy-Efficient Parallel Machines Scheduling, *IEEE Transactions on Cybernetics*, 2022, **52**(6), 5051-5063, doi: 10.1109/TCYB.2020.3026571.
[20] F. Zhao, Z. Wang, L. Wang, A reinforcement learning driven artificial bee colony algorithm for distributed heterogeneous No-wait flowshop scheduling problem with sequence-dependent setup times, *IEEE Transactions on Automation Science and Engineering*, 2023, **20**, 2305-2320, doi: 10.1109/TASE.2022.3212786.
[21] E. Ruschel, E. de Freitas Rocha Loures, E. A. P. Santos, Performance analysis and time prediction in manufacturing systems, *Computers & Industrial Engineering*, 2021, **151**, 106972, doi: 10.1016/j.cie.2020.106972.
[22] W. M. P. van der Aalst, M. H. Schonenberg, M. Song, Time prediction based on process mining, *Information Systems*, 2011, **36**, 450-475, doi: 10.1016/j.is.2010.09.001.
[23] H. Yamashiro, H. Nonaka, Estimation of processing time using machine learning and real factory data for optimization of parallel machine scheduling problem, *Operations Research Perspectives*, 2021, **8**, 100196, doi: 10.1016/j.orp.2021.100196.
[24] Wang. H, Sarker. B. R, Li. J, Li. J, Adaptive scheduling for assembly job shop with uncertain assembly times based on dual Q-learning, *International Journal of Production Research*, 2020, **59**(19), 5867–5883, doi: 10.1080/00207543.2020.1794075.
[25] S. M. Al-Selwi, M. F. Hassan, S. J. Abdulkadir, A. Muneer, E. H. Sumiea, A. Alqushaibi, M. G. Ragab, RNN-LSTM: from applications to modeling techniques and beyond: systematic review, *Journal of King Saud University - Computer and Information Sciences*, 2024, **36**, 102068, doi: 10.1016/j.jksuci.2024.102068.
[26] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, *IEEE transactions on evolutionary computation*, 2002, **6**, 182–197, doi: 10.1109/4235.996017.
[27] C. A. C. Coello, G. T. Pulido, M. S. Lechuga, Handling multiple objectives with particle swarm optimization, *IEEE Transactions on Evolutionary Computation*, 2004, **8**, 256-279,

doi: 10.1109/TEVC.2004.826067.
[28] M. Ali, P. Siarry, M. Pant, An efficient Differential Evolution based algorithm for solving multi-objective optimization problems, *European Journal of Operational Research*, 2012, **217**, 404-416, doi: 10.1016/j.ejor.2011.09.025.
[29] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: Improving the strength Pareto evolutionary algorithm, *TIK report*, 2001, **103**, doi: 10.3929/ethz-a-004284029.
[30] J. Behnamian, Decomposition based hybrid VNS–TS algorithm for distributed parallel factories scheduling with virtual corporation, *Computers & Operations Research*, 2014, **52**, 181-191, doi: 10.1016/j.cor.2013.11.017.