Journal of Project Management 11 (2026) 819–840

Contents lists available at GrowingScience

Journal of Project Management

homepage: www.GrowingScience.com/jpm

Dynamic flexible job shop scheduling using greedy actor–neural critic PPO reinforcement learning algorithm

Selva Kumar Chandrasekar[a*], Hariss Kumar Shanmugaprabu[a], Aswath Mani[a] and Nishanth Sami Raja Murugan[a]

[a]*Department of Mechanical Engineering, Thiagarajar College of Engineering, Madurai, India*

C H R O N I C L E

Article history:
Received February 12, 2026
Received in revised format March 14, 2026
Accepted April 19 2026
Available online
April 19 2026

Keywords:
Dynamic flexible job shop scheduling
Reinforcement learning
Makespan minimization
Energy consumption minimization
Machine utilization
Disruptions

A B S T R A C T

Dynamic Flexible Job Shop Scheduling (DFJSS) is a critical challenge in smart manufacturing due to dynamic job arrivals, machine breakdowns, routing flexibility, and multi-objective performance requirements. Conventional dispatching rules and metaheuristic algorithms often fail to provide adaptive real-time decisions. This paper proposes a Greedy Actor Neural Critic PPO based Reinforcement Learning (GANC-PPO-RL) framework to minimize makespan and total energy consumption. The problem is formulated as a Markov Decision Process, where machine availability, job progress, queue status, processing times, and energy levels define states, and job–machine assignments define actions. A multi-objective reward penalizes higher makespan and energy usage. The PPO algorithm ensures stable policy updates using a clipped objective. Results show makespan within ±2% of benchmarks for small instances and about 23% reduction for larger cases, with 16–20% lower makespan and 15–25% lower energy consumption compared to RL-QL.

© 2026 by the authors; licensee Growing Science, Canada.

1. Introduction

Manufacturing shop floors process a variety of jobs, each requiring multiple operations across different machines. The interaction between these jobs and limited machine availability means delays in one operation can affect overall progress, requiring careful production scheduling. This scheduling must optimize both the total manufacturing duration and the associated energy consumption to achieve overall efficiency (Sitahong et al., 2025). Shop floors often utilize a mix of machine types with different performance profiles. For instance, certain modern, high-capability machines prioritize speed at a higher energy cost, while more traditional or general-purpose machines favor energy efficiency with longer processing times. Within this context, Figure 1 illustrates a flexible job shop where each job can be processed through more than one feasible machine route. This scheduling flexibility requires the continuous negotiation of a fundamental trade-off, balancing the competing objectives of shorter production times and lower energy consumption (Chaudhry & Khan, 2016). The number of possible combinations of job-to-machine assignments grows rapidly with the number of jobs and machines, making schedule preparation a complex task. Though flexibility enables routing options, it does not account for the dynamic disruptions common in actual production, such as machine breakdowns, job cancellations, or the insertion of new orders (M. Zhang et al., 2021). Fig. 2 presents a dynamic job shop scenario in which a planned schedule is disrupted by several real-world events. A machine breakdown interrupts the processing of a job, forcing the reassignment of its remaining operations to other machines. Cancellation of previously scheduled job is a common disruption. This cancellation alters the expected machine load by leaving an unfinished workpiece in the system. Furthermore, the arrival of new job causes chaos. The urgent due date of job requires immediate integration into the existing workload. These disruptions — machine breakdown, job cancellation, and the insertion of new jobs — collectively invalidate the shop's schedule. They demand rapid and coordinated adjustments to maintain production viability (Rajabinasab & Mansour, 2011). However, responding effectively to such events becomes

* Corresponding author
E-mail address cskmech@tce.edu (S. K. Chandrasekar)

ISSN 2371-8374 (Online) - ISSN 2371-8366 (Print)
© 2026 by the authors; licensee Growing Science, Canada
doi: 10.5267/j.jpm.2026.4.009

significantly more complex when the system also possesses inherent routing flexibility. The integration of this flexibility with the need to manage dynamic disruption defines the core challenge of the dynamic flexible job shop scheduling problem (Destouet et al., 2025).

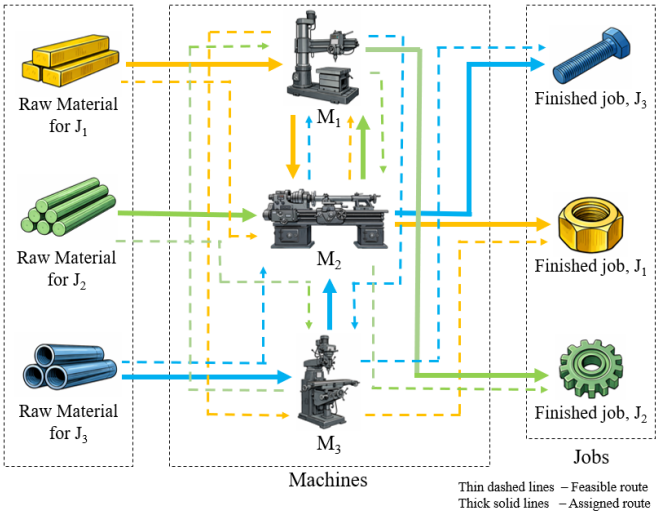

Fig. 1. Schematic illustration of a flexible job shop.

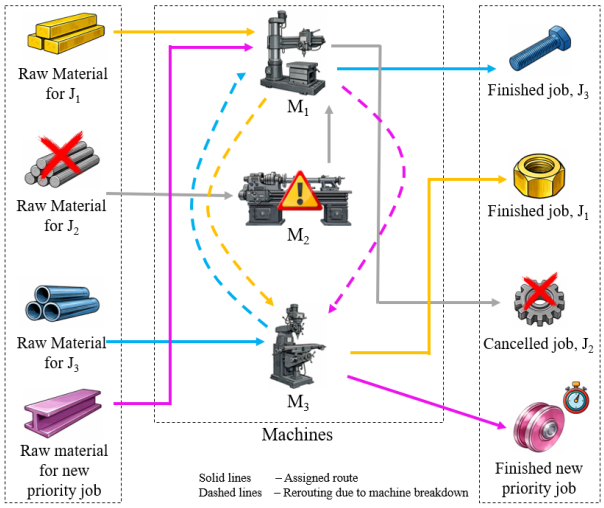

Fig. 2. Schematic illustration of a dynamic job shop.

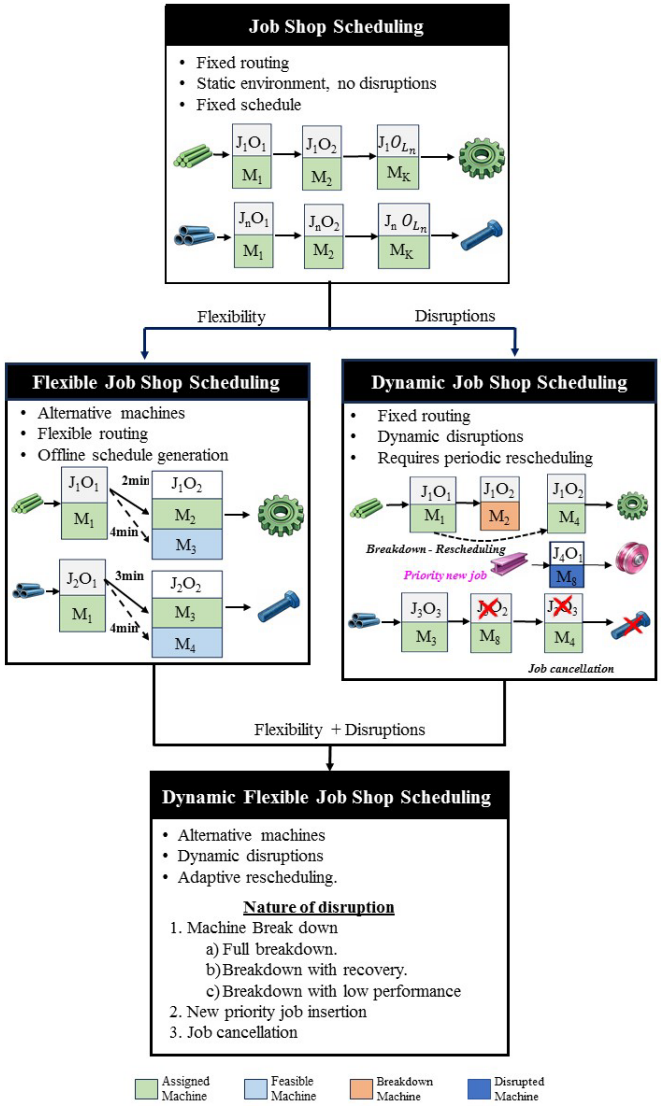

Fig. 3. Evolution of job shop scheduling into a dynamic flexible job shop environment

Fig. 3 summarizes the evolution of job shop scheduling into the dynamic flexible job shop scheduling environment, which simultaneously addresses two fundamental complexities: routing flexibility and real-time disruptions. In this integrated setting, each operation can be processed on multiple alternative machines, while the system must also respond to stochastic events such as machine breakdowns, job cancellations, and urgent order insertions (Shahgholi Zadeh et al., 2019). The core challenge is intensified by competing objectives, notably minimizing total completion time while also reducing overall energy consumption across a heterogeneous machine fleet. Consequently, the scheduler must continuously balance the efficiency gains from flexible routing against the robustness required handle dynamic disruptions. Managing this dual dynamic-flexibility interplay imposes a significant computational burden, especially as problem size increases. The number of possible machine assignments can be approximated as $Search\ Space \approx CC^{(x.y)}$, where N is the number of jobs, y is the average operations per job, and CC is the Combinatorial Coefficient that scales with flexibility. Specifically, $CC = \binom{K}{f}$, where K is the total number of machines and f is the average number of feasible machines per operation. This formulation, visually captured in Figure 4, demonstrates that the base of the exponent, CC, expands dramatically with even modest increase in f. For instance, increasing f from 1 to 5 can cause the base CC to grow by several orders of magnitude, which in turn triggers a combinatorial explosion in the search space.

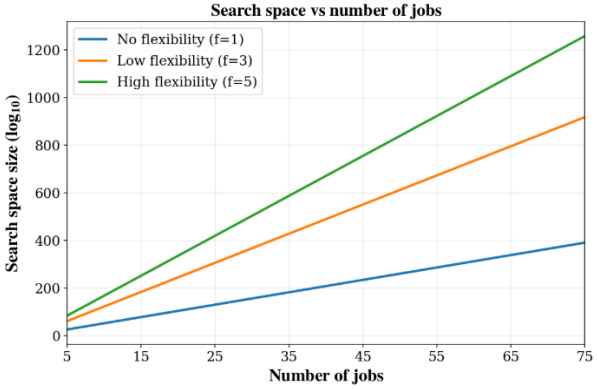

Fig. 4. Illustration of search space explosion as number of jobs and flexibility increases

The combinatorial explosion renders the problem computationally intractable at practical scales (Dauzère-Pérès et al., 2024). Consequently, enumerative methods are infeasible, and standard metaheuristics struggle with convergence in such a vast and dynamic search space. This necessitates a learning-based approach that can develop effective scheduling policies through interaction. Reinforcement learning (RL) is uniquely suited for this challenge, as it frames scheduling as a sequential decision-making process under uncertainty (Wang et al., 2021; Chung et al., 2025). Specifically, policy-based actor-critic methods, which balance exploration with stable policy updates, are a promising paradigm for this domain (J. Chang et al., 2022). However, effectively applying these advanced RL algorithms to the multi-objective, resource-constrained, and disruption-prone environment of the DFJSS remains a significant open challenge (X. Zhang & Zhu, 2025). This challenge is characterized by the need for efficient learning in a large-dimensional discrete action space while simultaneously optimizing for both time and energy, and maintaining robustness under dynamic disruptions. Addressing this gap forms the core motivation for this work. The key contributions for this work are,

1. We formulate a dynamic flexible job shop scheduling model that jointly minimizes makespan and energy consumption while explicitly accounting for real-world disruptions such as machine breakdowns, job cancellations, and urgent job insertions. This model reflects practical shop-floor behaviour where routing flexibility and dynamic uncertainty coexist and jointly influence scheduling quality.
2. We develop a Greedy Actor–Neural Critic Proximal Policy Optimization based Reinforcement Learning algorithm (GANC-PPO-RL) that combines greedy action selection with a neural network-based critic to guide policy improvement in high-dimensional scheduling spaces. The greedy actor accelerates decision making under routing alternatives, while the neural critic stabilizes learning and supports adaptive rescheduling under dynamic shop-floor conditions.
3. We perform an extensive empirical study that evaluates the proposed GANC-PPO-RL approach on both static flexible and dynamically disrupted scheduling scenarios, using performance indicators that reflect routing effectiveness, resource usage, and system robustness. The analysis covers production-focused measures such as makespan and energy usage as well as utilization- and disruption-oriented indicators relevant to DFJSS.
4. We demonstrate that the proposed reinforcement learning approach consistently outperforms established scheduling algorithms—GA, PSO, conventional RL—across static and dynamic instances. These results highlight the effectiveness of combining flexibility and dynamic disruptions within a unified RL-based scheduling framework.

The remainder of this article is organized as follows. Section 2 reviews recent research on flexible and dynamic job shop scheduling, along with reinforcement learning (RL)-based methodologies developed to address these problems. Section

3 presents the formulation of the dual-objective problem, considering makespan and energy consumption, including the associated constraints and performance metrics. Section 4 details the proposed GANC-PPO-RL algorithm and explains its implementation. Section 5 discusses the experimental results for various flexible and dynamic scenarios. Finally, Section 6 concludes the study by summarizing the key findings and outlining directions for future research.

2. Literature Review

The Flexible Job Shop Scheduling Problem (FJSSP) represents a critical and complex extension of the classical job shop, characterized by the routing flexibility of operations across multiple machines. Extensive research on this problem includes the comprehensive survey by Li et al., (2022), which details integrated models for production, maintenance, and transportation planning. Due to its NP-hard nature, achieving optimal solutions for medium or realistically sized problems with exact methods is intractable. This limitation necessitates the development of specialized methodological frameworks. Fattahi et al., (2007) formalized the fundamental division between hierarchical and integrated approaches, where hierarchical methods separate machine assignment from operation sequencing, and integrated methods solve them concurrently. Solving this problem therefore relies on advanced optimization techniques. For instance, Y. Zhang et al., (2019) developed an improved hybrid Particle Swarm Optimization algorithm to address multi-objective FJSSP efficiently. Similarly, a multi-objective evolutionary algorithm for robust scheduling under uncertainty was proposed by Shen et al., (2017). Other studies have focused on specific real-world constraints, such as energy-aware scheduling under time-of-use pricing, investigated by Park & Ham, (2022), and dual-resource constraints involving workers and machines, modeled by Vital-Soto et al., (2023). While these metaheuristic and mathematical programming approaches deliver high-quality solutions for static and deterministic problem formulations, they are inherently designed for offline optimization. Their computational expense and lack of adaptive policy learning limit their applicability in dynamic manufacturing environments where schedules are frequently invalidated by unforeseen disruptions.

To overcome the computational intractability of exact methods for NP-hard FJSSP instances, researchers have extensively developed evolutionary and heuristic techniques. These metaheuristics excel at navigating vast, complex solution spaces, delivering high-quality, near-optimal schedules within practical timeframes. A prominent trend is the hybridization of core algorithms to enhance performance. For instance, Sun et al., (2023) combined a genetic algorithm with variable neighborhood search to optimize makespan and machine workload balance. Evolutionary frameworks are also adapted for multi-objective and dynamic scenarios. Tang et al., (2016) and Rahmani & Ramezanian, (2016) employed an heuristic approach to solve the multi-objective problem of minimizing makespan and energy consumption simultaneously with disruptions. For handling multiple disruption types, including job replacement, Saouabi et al., (2024) introduced a two-level evolutionary algorithm. Yu et al., (2022) developed an adaptive multi-objective evolutionary algorithm for dynamic scheduling with order insertion uncertainty. Further innovations involve memetic structures that embed local search. Beyond time and cost, scheduling objectives now integrate essential production factors like energy consumption, modeled through machine speed-scaling (Salido et al., 2016), and product quality, addressed in models that account for abnormalities caused by machine faults (Zhu et al., 2020). Meanwhile Peng et al., (2025) proposed a two-stage efficient memetic algorithm for distributed scheduling with uncertain processing times. Even for static problems, enhancements like the hybrid Taguchi-genetic algorithm by H.-C. Chang et al., (2015) demonstrate superior convergence. The proven efficacy of these methods establishes them as the standard for large-scale problems. However, their offline nature and static formulation remain a critical gap in dynamic environments with frequent disruptions. This inherent limitation necessitates a shift toward intelligent, policy-learning approaches like reinforcement learning for adaptive, real-time scheduling.

The pivot towards intelligent, policy-learning frameworks has emerged to directly address the adaptability gap in conventional metaheuristics. Reinforcement Learning and its deep extensions have introduced a novel paradigm for adaptive scheduling, where agents learn generalizable policies through interaction with their environment. A reinforcement learning–based multi-agent framework was developed by Lopes Silva et al., (2019) to enhance adaptability and cooperation in routing and scheduling problems, showing that Q-learning–driven agents improve both individual and collective solution quality. This line of work naturally motivated the shift toward more advanced policy-learning frameworks to address increasing problem complexity. Early demonstrations by Han & Yang, (2021) confirmed that end-to-end Deep Reinforcement Learning models could surpass traditional dispatching rules. Subsequent research expanded the methodological toolkit by Zhao & Deng, (2023), who proposed actor–critic frameworks to enhance policy learning and stability. Proximal Policy Optimization was further improved by Liang et al., (2021) through the incorporation of specialized experience replay mechanisms to boost training efficiency and performance. A multi-action deep reinforcement learning framework was proposed by Lei et al., (2022) to address flexible job-shop scheduling by jointly learning operation selection and machine assignment policies using a graph neural network and a PPO-based training strategy. For complex large-scale problems, architectures integrating Graph Transformers have been introduced by W. Zhang et al., (2025) to capture intricate state dependencies. Furthermore, Deep Q-Network models have been applied to multi-objective scenarios with practical constraints like crane transport by Du et al., (2024). However, a survey of these intelligent approaches reveals a common limitation: while individual studies address specific dynamic events like new job insertion or uncertain processing times, there is a pronounced lack of comprehensive frameworks that simultaneously consider the full spectrum of dynamic disruptions—including machine breakdowns, new job arrivals, and job cancellations, —within a unified, multi-objective context for very large-scale instances. To bridge this

gap and deliver a robust solution for complex, real-world dynamic environments, this work advances and implements a Greedy Actor – Neural Critic PPO based Reinforcement Learning algorithm.

Table 1

Summary of the literature reviewed

Author	Method	Work Done		Dynamic Events		
		Makespan	Energy Consumption	Breakdown	New Job Insertion	Job Cancellation
Heuristics approaches						
H.-C. Chang et al., (2015)	Genetic algorithm enhanced with Taguchi method	✓	✗	✗	✗	✗
Tang et al., (2016)	Improved particle swarm optimization based multiobjective	✓	✓	✓	✓	✗
Rahmani & Ramezanian, (2016)	Dynamic Evolutionary Algorithm	✓	✗	✓	✓	✗
Salido et al., (2016)	Genetic Algorithm	✓	✓	✗	✗	✗
Zhu et al., (2020)	Improved Two-Stage Memetic Algorithm	✓	✓	✓	✗	✗
Sun et al (2023)	Hybrid genetic algorithm	✓	✗	✗	✗	✗
Saouabi et al., (2024)	Two-Level Evolutionary Algorithm	✓	✗	✓	✓	✓
Intelligent approaches						
Lopes Silva et al., (2019)	Multi-agent reinforcement learning based metaheuristic optimization	✓	✗	✗	✗	✗
Han & Yang, (2021)	Deep Reinforcement learning - Pointer Model	✓	✗	✗	✗	✗
Zhao & Deng, (2023)	Actor - Critic Driven Deep Q learning	✗	✗	✗	✓	✗
Liang et al., (2021)	Proximal Policy Optimization - Hybrid Prioritized Experience Replay	✓	✗	✗	✗	✗
Du et al., (2024)	Deep Q - network based scheduling	✓	✓	✗	✗	✗
W. Zhang et al., (2025)	PPO based Actor – Critic Deep Reinforcement Learning algorithm	✓	✗	✓	✗	✗
This work	Greedy Actor – Neural Critic PPO based Reinforcement Learning algorithm	✓	✓	✓	✓	✓

In summary, as consolidated in Table 1, the reviewed literature demonstrates substantial progress in addressing the DFJSS using metaheuristic, evolutionary, and reinforcement learning–based approaches. While metaheuristics remain effective for large-scale static problems, their offline nature limits adaptability in dynamic environments. Recent reinforcement learning methods introduce policy learning for adaptive scheduling, yet they typically address only limited disruption types or problem scales. This highlights a clear research gap for a unified, scalable, multi-objective framework capable of handling diverse real-world disruptions, motivating the approach proposed in this work.

3. Problem Description

The dynamic flexible job shop scheduling problem under disruptions is formulated here with the objectives of minimizing makespan and energy consumption. The job shop consists of N jobs, $J = \{J_1, J_2, .., J_n, .., J_N\}$, each comprising L_n that must be processed in a precedence constrained sequence on K available machines $M = \{M_1, M_2, \dots, M_k, \dots, M_K\}$. The processing time $P_{n,l,k}$ and energy consumption rate E_k for each operation depend on the selection machine M_k. The scheduling goal is to minimize both makespan C_{max} and total energy E, while ensuring robustness against dynamic disturbances such as

machine breakdowns, new job arrivals, and job cancellations. Key notations used in the formulation are summarized in Table 2.

Table 2
Notations used and their description

Notations	**Description**
Sets and Indices	
$J = \{J_1, J_2, .., J_n, .., J_N\}$	Set of jobs
$M = \{M_1, M_2, \dots, M_k, \dots, M_K\}$	Set of machines
$J_n O_l$	l^{th} operation of job n
n, l, k	Indices for job, operation, and machine
i	Global operation index, where $i = 1, \dots, I$ and $I = \sum_{n=1}^{N} L_n$
OL_n	Last operation index of job n
Time Variables	
$S_{n,l,k}$	Start time of operation $J_n O_l$ on machine k
$C_{n,l,k}$	Completion time of operation $J_n O_l$ on machine k
$P_{n,l,k}$	Processing time of operation $J_n O_l$ on machine k
R_{nl}	Release time of operation $J_n O_l$
C_{max}	Makespan
C'_{max}	Makespan after rescheduling
δ_t	Additional time due to disruptions
D_k	Downtime interval of machine k
Energy Variables	
E_k	Energy consumption rate of machine k
E	Total energy consumption
E'	Total energy consumption after rescheduling
δ_E	Additional energy due to disruptions
Decision Variable	
$A(k, i) \in \{0,1\}$	Binary assignment: 1 if operation i is processed on machine k
Performance Metrics	
U_k, UR_k	Utilization time and utilization rate of machine k
σ_U	Standard deviation of machine utilization
$\overline{UR}_k, \overline{IT}_k$	Average utilization and idle time of machine k
$\overline{C_n}, \overline{F_n}$	Average job completion time and flow time
Dynamic Event Parameters	
λ_e	Occurrence rate of dynamic event
b	Number of dynamic events
TR_e	Trigger interval time of dynamic event
U	Uniform random number $\in (0,1)$
$RI_{C_{max}}, RI_E$	Robustness Index for makespan and energy

3.1 Input parameters

The problem inputs consist of the processing time matrix P and the machine energy consumption rate E_K.

The processing time matrix P is of size $K \times I$, where K is the number of machine and $I = \sum_{n=1}^{N} L_n$ is the total number of operations across all jobs. Each entry $P_{k,i}$ denotes the processing time of operation i on machine k, as expressed in Eq. (1). Entries with $P_{k,i} = NA$ indicate that operation i cannot be processed on machine k, reflecting the flexibility level of the job shop. These entries are excluded from assignment during optimization.

$$\text{Processing time, } P_{k,i} = \begin{bmatrix} P_{1,1} & \cdots & P_{1,I} \\ \vdots & \ddots & \vdots \\ P_{K,1} & \cdots & P_{K,I} \end{bmatrix} \quad (1)$$

The energy consumption rate for each machine k is provided as input, as specified in Eq. (2). This rate defines the energy consumed per unit processing time by each machine. It directly influences the total energy usage of the schedule, which is a core optimization objective.

$$\text{Energy consumption, } E = \{E_1, E_2, \dots, E_k, \dots, E_K\} \quad (2)$$

3.2 Decision Variable

The decision variable is a binary assignment matrix *A,* where each element $A(k, i)$ indicates whether operation i is assigned to machine k. The matrix is mathematically represented in Equation (3).

$$A(k,i) = \begin{cases} 1, & if\ i^{th} operation\ \ of\ job\ n\ assigned\ to\ machine\ k \\ 0, & otherwise \end{cases} \quad (3)$$

Matrix A is a 2-dimensional matrix of size $(K \times I)$ that determines the allocation of each operation to the machines in M. Using the assignment matrix A, the completion time C_{nlk} for each operation is computed. The assignment matrix A thus fully defines the scheduling decisions, from which all makespan and energy outcomes are derived.

3.3 Objectives and constraints

The multi-objective nature of the problem requires simultaneous minimization of the makespan C_{max} and total energy consumption E. These objectives are inherently conflicting: reducing makespan often requires utilizing faster, higher-energy machines, whereas lowering energy consumption may prioritize slower, energy-efficient machines, thereby extending the schedule duration.

Makespan Minimization

The makespan is determined by the completion times $C_{n,l,k}$, which depend on the machine assignment $A(k,i)$ and the processing time $P_{n,l,k}$.

$$minimize\ C_{max} = \max(c_{nlk}) \tag{4}$$

where,

$$C_{nlk} = S_{nlk} + P_{nlk} \cdot A(k,i) \tag{5}$$

$$S_{nlk} = \max\left(C_{n,l-1,k}, C_{l'k}, R_{nl}\right) \tag{6}$$

$$S_{n'l} \geq R_{n'l}. \tag{7}$$

The completion time C_{nlk} is calculated by Equ. (5) which sums the start time S_{nlk} and the product of processing time P_{nlk} and the assignment variable $A(k,i)$. Constraint (6) enforces that the start time S_{nlk} must be at least the maximum of the completion time of the job's preceding operation $C_{n,l-1,k}$, latest completion time on the same machine $C_{l'k}$, and release time R_{nl} of the job if it is due to dynamic disruption. This ensures operation precedence, machine availability, and prevents scheduling before job release. Constraint (7) explicitly requires that the start time $S_{n'l}$ of newly inserted job is not earlier than its release time $R_{n'l}$, thereby ensuring temporal admissibility under dynamic job arrivals.

Energy Consumption Minimization

The total energy E is calculated as the sum of products of $A(k,i).P_{n,l,k}.E_k$, linking each operation assignment to its machine-specific energy rate. The objective is to minimize E by preferentially assigning operations to machines with lower energy consumption, as expressed in Eq. (8)

$$min\ E = \sum_{k \in M} \sum_{i \in JO} A(k,i) \cdot P_{nlk} \cdot E_k \tag{8}$$

subject to,

$$\sum A(k,i) = 1 \tag{9}$$

$$A(k,i) \in \{0,1\} \tag{10}$$

$$S_{n,l+1,k} \geq S_{nlk} + P_{nlk}.A(k,i) \tag{11}$$

$$S_{nlk} \geq D_k^{end}\ or\ S_{nlk} + P_{nlk} \cdot A(k,i) \leq D_k^{start} \tag{12}$$

$$S_{n'lk} + P_{n'lk}.A(k,i) \leq C_{max} \tag{13}$$

Constraint (9) ensures each operation i is assigned to exactly one machine k, guaranteeing that all operations are scheduled. Constraint (10) enforces the binary nature of the assignment variable $A(k,i)$, where a value of 1 indicates assignment and 0 indicates non-assignment. Constraint (11) maintains operation precedence within each job by ensuring that the start time of a subsequent operation $S_{n,l+1,k}$ is no earlier than the completion time of its predecessor $S_{nlk} + P_{nlk}.A(k,i)$. Constraint (12) dictates the machine cannot be assigned with operation during downtime $[D_k^{start},\ D_k^{end}]$. Constraint (13) ensures new job is included in makespan C_{max}, thereby integrating dynamic arrivals into the schedule.

To simultaneously address the competing objectives of makespan and energy minimization, a single weighted objective function is formulated as shown in Eq. (14).

$$min\ f = w_1 C_{max} + w_2\ E \tag{14}$$

To investigate the influence of relative between makespan and energy consumption, three weighting schemes are considered in the weighted multi-objective formulation. The weights are selected such that $w_1 + w_2 = 1$, while shifting the emphasis between the two objectives:

- Makespan focused configuration: $w_1 = 0.8$, $w_2 = 0.2$
- Balanced configuration: $w_1 = 0.5$, $w_2 = 0.5$
- Energy focused configuration: $w_1 = 0.2$, $w_2 = 0.8$

This weighted formulation allows the prioritization of either objective or seek a balanced compromise, facilitating analysis under different operational preferences.

3.4 Performance metrics for flexible scheduling

To comprehensively evaluate the performance of the proposed scheduling approach under flexible machine assignments, a set of quantitative metrics is introduced. These metrics assess resource utilization efficiency, load balancing effectiveness, and operational timeliness, providing a multi-faceted view of schedule quality. Specifically, average machine utilization rate, standard deviation of machine utilization, average machine idle time, and average job completion time are examined. Each metric is formally defined and interpreted in the context of flexible job shop scheduling, enabling systematic comparison and analysis of scheduling outcomes.

3.4.1. Average Machine Utilization Rate

The average machine utilization rate $\overline{UR}_k$ is the mean utilization percentage across all machines, this measures how well all the machines are utilized in a schedule without much idle time. The value of the average utilization rate ranges from 0 to 1, where $0 \leq \overline{UR}_k \leq 0.4$ indicates low utilization, $0.4 \leq \overline{UR}_k \leq 0.7$ corresponds to moderate utilization, and $0.7 \leq \overline{UR}_k \leq 1$ signifies high machine utilization. Equation (15) mathematically expresses the average machine utilization rate, $\overline{UR}_k$.

$$\overline{UR}_k = \frac{\sum_k UR_k}{k} \tag{15}$$

where,

$$UR_k = \frac{U_k}{\tau_k} \tag{16}$$

$$U_k = \sum_{i \in JO} A(k,i).P_{i,k} \tag{17}$$

$$\tau_k = C_{max} - D_k \tag{18}$$

The utilization rate UR_k in Eq. (16) quantifies the proportion of the time machine k is active relative to its total available time τ_k. The machine utilization time U_k is the total time the machine M_k is processing its assigned jobs, as shown in Eq. (17). It is the sum of processing time $P_{i,k}$ of all the operations assigned on a given machine k. The total available time τ_k, defined in Eq. (18), is the duration during which machine k is operational, defined as makespan C_{max} minus the down time D_k of the machine.

3.4.2. Standard Deviation of Machine Utilization

The standard deviation of machine utilization σ_U, gives the distribution of work load across machines, which is an important performance metric to be considered in a scheduling environment to prevent excessive workload on few machines. It is calculated as shown in Eq. (19).

$$\sigma_U = \sqrt{\frac{1}{|K|}\sum_{k \epsilon M}(UR_k - \overline{UR_k})^2} \tag{19}$$

Low standard deviation of machine utilization σ_U $(0 \leq \sigma_U \leq 0.1)$ reflects excellent load balance with no dominant bottleneck machines. Moderate σ_U $(0.1 \leq \sigma_U \leq 0.2)$ shows acceptable imbalance and high σ_U $(0.2 \leq \sigma_U \leq 0.5)$ shows highly inefficient schedule with poor balance. The theoretical maximum of 0.5 corresponds to the extreme case where half the machines are fully utilized and the remainder are idle.

3.4.3. Average Machine idle time

The average machine idle time $\overline{IT}_k$, defined in Eq. (2), measures the total idle duration across all machines in the job shop. Minimizing this metric is indicative of effective scheduling and improved resource utilization.

$$\overline{IT}_k = \frac{IT_k}{K} \tag{20}$$

where,

$$IT_k = C_{max} - D_k - U_k \tag{21}$$

In Eq. (21), IT_k represents the idle time of machine k calculated as the makespan $C_{\max}$ minus the downtime D_k and the total processing time U_k of the machine. This formulation quantifies the duration during which the machine is available but not engaged in any operation.

3.4.4. Average Job Completion Time

The average job completion time $\bar{C}_n$, defined in Eq. (22), represents the mean duration required to complete jobs from their release into the job shop. Minimizing this metric is critical for reducing work-in-process inventory and buffer requirements, thereby enhancing system responsiveness.

$$\bar{C}_n = \frac{1}{|N|}\sum_{n} C_{L_n} \tag{22}$$

Here, C_{L_n} denotes the completion time of the last operation of job n, and $|N|$ is the total number of jobs.

3.5 Dynamic event modelling and robustness assessment

To evaluate scheduling performance under realistic uncertainty, this section presents a stochastic framework for modeling dynamic disruptions and measuring their impact on schedule robustness. The framework includes a Poisson-based trigger mechanism to simulate random events, formulations that update makespan and energy after rescheduling, and key performance metrics such as the Robustness Index and Average Job Flow Time. These components collectively assess system resilience and responsiveness in dynamic environments.

3.5.1 Dynamic event framework

In DFJSS, the schedule is subject to unpredictable disruptions that necessitate real-time adaptation. Three primary categories of dynamic events are modelled, each characterized by type, timing, and severity:

1. Machine breakdown:
   a. *Full breakdowns*: Machines become fully unavailable for a duration.
   b. *Partial breakdowns*: Machines remain operational but with reduced speed and increased energy consumption.
   c. *Recovery scenarios*: Machines may resume operation after a specified downtime.
2. Job cancellation:
   Certain jobs may be cancelled during execution due to unavoidable circumstances, thereby reducing the remaining workload and altering the schedule dynamics.
3. New job insertions:
   New jobs are introduced dynamically, requiring integration into the ongoing schedule.

These disturbances invalidate the initial schedule and require rescheduling to restore feasibility and maintain performance. To systematically represent these real-world uncertainties, a probabilistic event modelling framework is adopted, where disruptions are characterized by their frequency, timing, and impact on schedule metrics. This stochastic approach ensures that scheduling algorithms respond to realistic, non-deterministic scenarios rather than assuming static or fully predictable conditions.

3.5.2 Stochastic Event Trigger Using Poisson Process

Dynamic events, including machine breakdowns, job cancellations, and new job arrivals, are integrated into the scheduling framework through a Poisson-based stochastic trigger mechanism. This approach models the random occurrence of disruptions over the scheduling horizon. The probability of b such dynamic events occurring within a given time horizon T is expressed in Eq. (25).

$$\text{Probability, } P(x = b) = \frac{e^{-\lambda_e T}(\lambda_e T)^b}{b!} \tag{25}$$

Here λ_e denotes the average occurrence rate of dynamic events. The inter-arrival time between successive disruption events, denoted by TR_e, follows an exponential distribution. This is expressed in Equation (26).

$$\text{Trigger interval time, } TR_e = -\frac{1}{\lambda_e} ln(U) \tag{26}$$

Here U is a uniform random variable on the interval $(0, 1)$. This formulation allows the generation of event sequences with realistic temporal randomness, providing a robust basis for evaluating the scheduler's responsiveness and resilience in dynamic environments.

3.5.3 Rescheduling impact and robustness index

When a dynamic disruption triggered by the Poisson process occurs, the original schedule must be adjusted to accommodate the changed conditions. This rescheduling affects both the makespan and total energy consumption. The updated makespan after rescheduling, denoted by C'_{max}, incorporates any additional time delay δ_t caused by disruptions such as machine downtime or newly inserted jobs. The revised makespan is expressed in Eq. (25).

$$C'_{max} = C_{max} + \delta_t \tag{25}$$

Similarly, the total energy consumption after rescheduling, E', includes any incremental energy δ_E resulting from disruptions, for example, increased energy consumption due to machine slowdowns or the processing of additional jobs. The revised energy consumption is expressed in Equation (26).

$$E' = E + \delta_E \tag{26}$$

To evaluate how effectively the schedule withstands disruptions, a Robustness Index RI is introduced. For each objective, RI is defined as the ratio of the original performance to the post-disruption performance. robustness indices for makespan $RI_{C_{max}}$ and energy RI_E are given in Eq. (27) and Eq. (28), respectively.

$$RI_{C_{max}} = \frac{C_{max}}{C'_{max}} \tag{27}$$

$$RI_E = \frac{E}{E'} \tag{28}$$

Values of RI closer to 1 indicate minimal degradation in makespan or energy consumption after a disruption. This reflects strong schedule stability, which in an industrial context reduces downtime, maintains throughput, and supports consistent energy efficiency under unpredictable operating conditions.

3.5.4 Average job flow time

The average job flow time $\bar{F}$ measures the mean duration jobs spend in the system from release to completion, capturing both processing and waiting times. It is calculated as shown in Eq. (29).

$$\bar{F} = \frac{1}{N}\sum_{n=1}^{N} F_n \tag{29}$$

where, F_n is the flow time of the job n, defined in Eq. (30).

$$F_n = C_n - R_n \tag{30}$$

Here, C_n is the completion time and R_n is the release time for job n. For jobs available from the start of the schedule $(R_n = 0)$, flow time equals completion time. For jobs inserted dynamically after the schedule begins $(R_n > 0)$, flow time measures the total duration from insertion to completion, capturing the system's responsiveness to real-time arrivals. Minimizing $\overline{F}$ is critical for reducing work-in-process inventory and improving delivery responsiveness in dynamic job shop environments.

3.6 Assumptions

The following assumptions are made to define the problem scope and ensure model consistency. All machines are available at time zero. Operations are non-pre-emptive and may only be interrupted by machine breakdown. Each job's operations must follow a strict sequence. Processing times $P_{n,l,k}$ and machine energy rate E_k are known in advance. The duration of machine downtime D_k is known in advance, though its occurrence timing is uncertain and treated as part of the dynamic event model. Transit time between machines is negligible.

4. Greedy Actor–Neural Critic Proximal Policy Optimization based Reinforcement Learning algorithm

In recent years, reinforcement learning has evolved from solving low-dimensional control problems to addressing large-scale combinatorial optimization tasks with extremely high-dimensional and discrete search spaces. Classical scheduling problems, particularly the Dynamic Flexible Job Shop Scheduling Problem, are characterized by combinatorial explosion due to routing flexibility, machine alternatives, and dynamic events such as job arrivals and machine disturbances. Traditional exact and

metaheuristic approaches often struggle to maintain scalability and real-time responsiveness under such conditions. This has motivated a shift towards learning-based approaches, where reinforcement learning is increasingly combined with heuristics, search strategies, and domain knowledge to improve convergence speed, stability, and solution quality. Hybrid frameworks that integrate greedy rules, neural critics, and policy optimization have shown promising performance in navigating complex scheduling landscapes while retaining computational efficiency.

Building on this progress, this section presents a Greedy Actor–Neural Critic Proximal Policy Optimization based reinforcement learning framework tailored for the Dynamic Flexible Job Shop environment. The overall workflow of the proposed methodology is illustrated in Fig. 5, which shows how the scheduling environment is initialized, how actions are selected through a dual-actor mechanism, and how decisions are evaluated using a neural critic within a PPO-based learning loop. The approach begins with formal definition of the Markov Decision Process, construction of the state representation, and normalization of input features to ensure stable learning. Feasible operations are then extracted and evaluated using a combination of policy-driven and greedy strategies, enabling efficient navigation of the large combinatorial action space. The critic network estimates future penalties to guide decision making and improve policy stability, while episode-wise evaluation and advantage computation drive the PPO rollout updates. The learning process continues iteratively until convergence criteria are satisfied, after which the final assignment matrix is generated. Together, these components form an integrated learning framework that balances exploration and exploitation while maintaining computational tractability in dynamic scheduling scenarios.

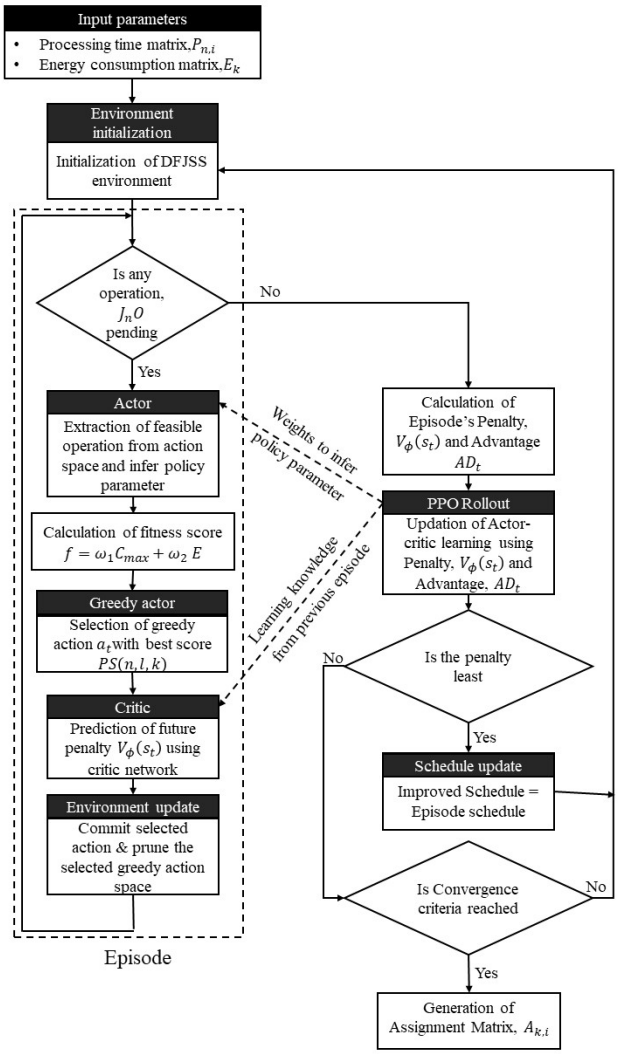

Fig. 5. Flowchart of Greedy Actor–Neural Critic PPO-based reinforcement learning framework

4.1 Environment Initialization

The DFJSS is formulated as a Markov Decision Process (MDP), as expressed in Eq. (31), in which the shop floor state evolves as scheduling decisions are made.

$$MDP = \langle S, A, \Pi, R, \gamma \rangle \tag{31}$$

where $s \in S$ is the system state, $a \in A$ is the scheduling action, $\Pi(s'|s,a)$ is the transition from state s to a new state s' by action a, $R(s,a)$ is the reward function, $\gamma \in (0,1)$ is the discount factor

At decision time t, the state is encoded as a feature vector capturing job progress, machine availability, and energy usage. The state at time t is given in Eq. (32).

$$s_t = [\eta_n(t), a_\kappa(t), \bar{p}_n(t), u_k(t), E(t)] \tag{32}$$

where $\eta_n(t)$ denotes the completion ratio of job n, $a_k(t)$ is the next available time of machine k, $\bar{p}_n(t)$is the minimum remaining processing time of job n, $u_k(t)$represents the utilization of machine k, and $E(t)$is the accumulated energy consumption.

Eq. (33) represents the computation of immediate reward. The reward function plays a critical role in guiding the reinforcement learning agent toward desirable scheduling behaviour. In the proposed framework, the immediate reward reflects the incremental impact of a scheduling decision on both production completion time and energy consumption. Specifically, the reward penalizes increases in makespan and energy usage using a weighted formulation.

$$\text{Immediate reward } r_t = -(\omega_1 \Delta C_{max} + \omega_2\, \Delta E) \tag{33}$$

where ΔC_{max} is the increment in makespan and ΔE is the increment in energy consumption, and ω_1, ω_2 control the relative importance of production efficiency and energy efficiency. This formulation allows the scheduling agent to balance conflicting objectives dynamically and encourages decisions that simultaneously reduce production delay and energy usage. Furthermore, by using incremental reward feedback, the PPO agent can capture the long-term scheduling consequences of each decision during policy learning.

In implementation, the state vector is constructed by concatenating global, per-job, and per-machine features, resulting in a variable-dimensional observation. To ensure numerical stability during neural network training, the raw state vector is normalized online using running statistics within the PPO framework. The normalised state vector is given in Eq. (34).

$$\text{Normalised state vector } \hat{s}_t = \frac{s_t - \mu_t}{\sqrt{\sigma_t^2 + \varepsilon}} \tag{34}$$

where μ_t and σ_t^2 are the running mean and variance of the state components, and ϵ is a small constant to avoid division by zero.

4.2 Greedy Actor Mechanism for Action Selection

In the proposed framework, the scheduling action at time t is defined as the assignment of the l-th operation of job n to machine k, denoted as $a_t = (n, l, k)$. Rather than directly selecting from this combinatorial action space, the agent employs a dual-actor mechanism: a learned PPO actor that outputs adaptive scheduling priorities, and a deterministic greedy actor that interprets these priorities to select concrete job–machine assignments.

The PPO actor, parameterized by θ, is a neural network that maps the observed state s_t to a continuous vector of heuristic weights w, as given in Eq. (35).

$$\omega = [\omega_{cp}, \omega_d, \omega_l, \omega_{wf}, \omega_e] \tag{35}$$

The critical path weight 'ω_{cp}' prioritizes jobs with long remaining operation (critical path), Delay weight 'ω_d' penalizes waiting and delays, Load balancing weight 'ω_l' promotes load balancing across machines, Waiting factor weight 'ω_{wf}' accounts for future congestion through look ahead waiting effects, and Energy weight 'ω_e' biases scheduling toward energy efficient assignments. Together, these weights enable the PPO agent to balance makespan, delay, workload distribution, and energy consumption. These weights are generated by a neural network policy whose trainable parameters are denoted by θ. The mapping from the observed state $s_t$to the heuristic weights is probabilistic and is modeled using a multivariate Normal Gaussian (NG) distribution as shown in Eq. (36).

$$\Pi_\theta(\omega|s) = NG(\mu_\theta(s), \Sigma_\theta) \tag{36}$$

where $\mu_\theta(s)$ is the mean vector produced by the actor network for state s, and Σ_θ is the covariance matrix that determines the variance of each heuristic weight and therefore controls the amount of exploration. Once ω is obtained, the greedy actor evaluates every feasible assignment (n, l, k) using a rule-based scoring function f_h, parameterized by the learned weights.

$$PS(n, l, k) = f_h(CP, D, L, WF, E|\, \omega) \tag{37}$$

Using Eq. (37), for each feasible assignment (n, l, k), a priority score $PS(n, l, k)$ is computed using a weighted combination of scheduling-relevant features. This score quantifies the relative suitability of the assignment under the current learned priorities ω. Here CP denotes remaining critical path length, D represents delay terms, L is machine load,WF is a look ahead waiting factor, and E is energy cost. The function f_{h} follows a rule-based structure to distinguish immediate and delayed scheduling decisions. Finally, the greedy actor selects the assignment with the minimum priority score, executing a deterministic scheduling decision, as expressed in Equation (38).

$$a_t = \arg\min PS(n, l, k) \tag{38}$$

Thus, the PPO actor learns how to weight competing scheduling objectives in response to dynamic shop-floor conditions, while the greedy actor efficiently enacts these priorities by selecting the best immediate assignment from the feasible set. This decoupling enables stable gradient-based learning in a continuous weight space while remaining computationally tractable in large, discrete action environments.

4.3 Critic Network for Long-Term Value Estimation

The critic network as illustrated in Fig. 6 is used to evaluate the long-term effect of partially constructed schedule. After the greedy actor generates a scheduling decision, the environment moves to a new state s_t. This new state is fed to the critic network. The critic is a neural network consisting of fully connected hidden layers with ReLU activations in the hidden layers. It receives the normalized state vector s_t as input and outputs a scalar estimate $V_\phi(s_t)$ representing the predicted discounted cumulative future reward from state s_t. This scalar estimate is mathematically expressed in Equation (39)

$$V_\phi(s_t) = \sum_{l=0}^{\infty} \gamma^\tau r_{t+\tau} \tag{39}$$

Here $r_{t+\tau}$ denotes the future rewards, γ is the discount factor, and ϕ indicates the parameters of the critic network. Following the execution of action a_t, the environment yields an immediate reward r_t and transitions to state s_{t+1}. Using this transition, as shown in Eq. (40), a temporal-difference (TD) target is constructed.

$$\hat{v}_t = r_t + \gamma V_\phi(s_{t+1}) \tag{40}$$

This target provides a bootstrap estimate of the true value of s_t. During training, the critic is optimized by minimizing the squared prediction error (SPE) between its prediction and the TD target as expressed in Eq. (41).

$$SPE = (V_\phi(s_t) - \hat{v}_t)^2 \tag{41}$$

Minimizing this loss enables the critic to accurately assess the long-term quality of scheduling decisions, which in turn supports stable and informed policy updates during PPO rollouts.

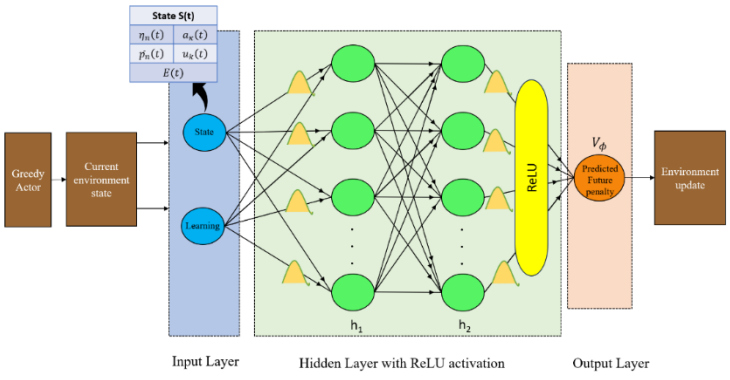

Fig. 6. Flowchart of Critic Network

4.4 Episode Evaluation and PPO Rollout

After the greedy actor selects and executes scheduling action a_t at time step t, the environment transitions to state s_{t+1}. The quality of this decision is immediately evaluated through the reward signal defined in Eq. (33), which reflects both makespan and energy usage.

To assess whether the taken action performed better or worse than expected, the temporal-difference error (TDE) is computed using the critic's value estimates as shown in Eq. (42).

$$\text{Temporal difference error, } TDE = r_t + \gamma V(s_{t+1}) - V(s_t) \tag{42}$$

where $V(s)$denotes the critic's predicted return from state s, and $\gamma$is the discount factor that controls the importance of future rewards. The TDE quantifies the discrepancy between the observed outcome and the critic's prediction, serving as a one-step advantage estimate. As a single decision can affect the schedule far into the future, the PPO algorithm accumulates these temporal difference errors over time using a recursive formulation known as the advantage function. The advantage at time t is computed as expressed in Eq. (43).

$$\text{Advantage, } AD_t = TDE + \gamma.\lambda.AD_t \tag{43}$$

where $\lambda \in (0,1)$ is an attenuation parameter controlling the bias–variance trade-off. This recursive formulation propagates future evaluation errors backward, ensuring each decision is credited or penalized based on its long-term impact.

As the agent interacts with the DFJSS environment, it collects a sequence of state–action–reward–value tuples $(s_t, \omega_t, r_t, V_\phi(s_t))$ while the policy remains unchanged. This sequence is termed a rollout. The trajectory buffer stores the rollout data until a terminal condition is reached, such as the completion of all job operations, as shown in Eq. (44).

$$\text{Trajectory Buffer, } TB = \{s_t, \omega_t, r_t, V_\phi(s_t)\}_{t=0}^{T-1} \tag{44}$$

This rollout buffer stores the full trajectory of the scheduling process until all operations are assigned or a terminal condition is reached. After the rollout is completed, PPO updates the actor policy by comparing the likelihood of the sampled heuristic weights under the current and previous policies via the probability ratio, shown in Eq. (45).

$$\text{PPO probability ratio, } r_t(\theta) = \frac{\Pi_\theta(\omega_t|s_t)}{\Pi_{\theta old}(\omega_t|s_t)} \tag{45}$$

This ratio measures the extent of policy change for the same state–action pair. Using this ratio $r_t(\theta)$ and the advantage value AD_t, PPO constructs a clipped objective to prevent excessively large policy updates that could destabilize learning. The clipped PPO objective is defined in Eq. (46).

$$\text{Clipped PPO objective, } L_{clip} = \min\left(r_t AD_t, clip(r_t, 1-\varepsilon, 1+\varepsilon)AD_t\right. \tag{46}$$

where ε is the clipping threshold. This objective ensures monotonic policy improvement while maintaining training stability.

Through repeated rollouts and PPO updates, the actor gradually learns how to generate heuristic weight vectors that lead to schedules with shorter makespan and lower energy consumption.

4.5 Convergence criteria and Generation of Assignment Matrix

During PPO training, the quality of the learned scheduling policy is continuously monitored using the average episode reward, which reflects the combined effect of makespan reduction and energy savings. To mitigate episodic noise and ensure stable convergence, a moving average over a window of W episodes is maintained. Convergence is declared when the change in this smoothed average reward remains within a small tolerance ε for P consecutive training iterations, as shown in Eq. (47).

$$|\bar{R}(t) - \bar{R}(t-P)| < \varepsilon \tag{47}$$

where $\bar{R}(t)$ denotes the average reward over episodes $[t-W+1, t]$ and $\bar{R}(t-P)$ is the corresponding average P iterations earlier. This condition indicates that further training will not significantly improve scheduling performance and that the policy has reached a stable solution. By repeatedly applying this process from the initial state until all operations are assigned, a complete production schedule is generated. The final result of this process is expressed in the form of a job machine assignment matrix A, whose elements are defined as in Eq. (3). This matrix provides a compact and explicit representation of the final schedule produced by the PPO learned heuristic. Each entry directly indicates which machine processes each

operation, making the output suitable for implementation on the shop floor or for further optimization and analysis. Thus, PPO training determines how scheduling criteria should be weighted, convergence ensures that these weights are reliable, and the assignment matrix translates this learned policy into an executable production plan.

5 Results and Discussion

This work has proposed a Greedy Actor Neural Critic Proximal Policy Optimization-based Reinforcement Learning (GANC-PPO-RL) framework for solving the Dynamic Flexible Job Shop Scheduling Problem under realistic shop floor disturbances. The proposed model explicitly incorporates machine breakdowns, breakdowns with recovery, partial breakdowns with reduced capacity, new job insertions, and job cancellations, all triggered through a Poisson-based stochastic event mechanism. These dynamic events continuously disrupt the original schedule, requiring real-time rescheduling while preserving routing flexibility and multi-objective optimization of makespan and energy consumption. The effectiveness of the proposed framework is evaluated on large DFJSS instances with intensive dynamic events, simulated on a computing system with AMD Ryzen 7 4800H CPU (2.90 GHz), 16 GB RAM, RTX 3050Ti GPU. To validate the effectiveness of the proposed GANC-PPO-RL framework, its performance is systematically compared with representative learning-based and meta-heuristic scheduling algorithms, namely Q-learning-based reinforcement learning (RL-QL), Particle Swarm Optimization (PSO), and the Genetic Algorithm (GA). All competing methods were implemented under identical experimental conditions using carefully selected parameter settings to ensure a fair and meaningful comparison. The complete set of algorithmic parameters for the proposed approach and the benchmark methods is summarized in Table 3.

Table 3
Algorithm parameters

Algorithm	Parameters
Proposed GANC-PPO-RL	Learning rate, $\alpha = 3 \times 10^{-4}$ Discount factor, $\gamma = 0.995$ PPO clipping parameter, $\varepsilon = 0.15$ Heuristic weight bounds: $w_{cp} \in [-30, 0]$ $w_{delay} \in [0, 20]$ $w_{load} \in [0, 20]$ $w_{wf} \in [0, 10]$ $w_{energy} \in [0.1, 50]$
Reinforcement Learning (Q-Learning)	Learning rate, $\alpha = 0.1$ Discount factor, $\gamma = 0.95$ Exploration strategy – Epsilon Greedy Decay rate = 0.995 Episodes = 3000
Particle Swarm Optimisation	Swarm size = 30 Inertia weight, $w = 0.4$ Cognitive coefficient, $c_1 = 0.8$ Social coefficient, $c_2 = 0.8$ Maximum iterations = 100
Genetic algorithm	Population size = 50 Crossover probability = 0.8 Mutation probability = 0.2 Selection method: Tournament selection Number of generations: 100

The proposed GANC-PPO-RL framework was trained using a learning rate of $\alpha = 3 \times 10^{-4}$, a discount factor of $\gamma = 0.995$, and a PPO clipping parameter of $\varepsilon = 0.15$, which together ensure stable and long-term oriented policy learning in the dynamic scheduling environment. The PPO actor learns continuous heuristic weights within bounded ranges that control scheduling behavior, where the critical-path weight $w_{cp} \in [-30,0]$ prioritizes long remaining job paths, the delay weight $w_{delay} \in [0,20]$ penalizes waiting, the load weight $w_{load} \in [0,20]$ promotes machine balance, the waiting-factor weight $w_{wf} \in [0,10]$ captures future congestion, and the energy weight $w_{energy} \in [0.1,50]$ biases decisions toward energy-efficient machines, allowing the greedy actor to convert PPO learned priorities into effective job–machine assignments under dynamic disruptions. For comparison, a Q-learning method with standard learning and discount rates, epsilon-greedy exploration, and a decaying epsilon schedule was implemented. PSO used a swarm size of 30 with Inertia weight, $w = 0.4$, Cognitive coefficient, $c_1 = 0.8$, Social coefficient, $c_2 = 0.8$, and 100 iterations, while GA employed a population of 50, crossover probability 0.8, mutation probability 0.2, tournament selection (size 3), and 100 generations. These baselines represent heuristic and learning-based approaches, whereas the proposed GANC-PPO-RL framework achieves superior adaptability, stability, and performance through its neural critic and greedy actor in large-scale dynamic flexible job shop scheduling.

5.1 Performance on Static Flexible Job Shop Instances

For the flexible job shop scheduling problem without dynamic disruptions, the proposed GANC-PPO-RL framework was evaluated on eight benchmark instances. Six of these are well-established datasets from the literature, while two large-scale

instances were generated in this study to assess scalability. The selected benchmarks span a wide range of problem sizes, with the number of jobs, operations, and machines increasing progressively from small to very large configurations, as summarized in Table 4.

Table 4
Description and makespan performance of flexible job shop instances: literature benchmarks vs. proposed GANC-PPO-RL.

Flexible Instance ID	Reference	Instance description			Makespan from Literature	Makespan from GANC-PPO-RL	Improvement (%)
		Jobs, N	Operations, I	Machines, K			
1	Fattahi et al., (2007b)	12	48	8	1384	1388	-0.29
2	Chandrasekar et al., (2025)	15	90	25	51	52	-1.9
3	Hurink et al., (1994)	15	225	15	955	955	0
4	Brandimarte, (1993)	30	284	15	383	377	+2.87
5	Behnke & Geiger, (2012)	99	495	60	537	410	+23.65
6	Behnke & Geiger, (2012)	100	500	60	532	409	+23.12
7	Proposed dataset (Zenodo)	60	540	25	–	261	–
8	Proposed dataset (Zenodo)	80	878	35	–	397	–

Note: The complete datasets for Instances 7 and 8 are too large to be included in this manuscript and are publicly available at Zenodo (DOI: https://doi.org/10.5281/zenodo.18865577)

For smaller benchmark problems, the proposed GANC-PPO-RL framework achieves makespan values that are highly consistent with those reported in the literature, with deviations remaining within ±2% for Fattahi et al., (2007b) (−0.29%) and Chandrasekar et al., (2025) (−1.9%), and an exact match for Hurink et al., (1994). This confirms the correctness and stability of the learning framework in low-dimensional scheduling scenarios. As the problem size increases, however, GANC-PPO-RL demonstrates a clear and growing advantage, achieving substantial makespan reductions of 23.65% and 23.12% for the Behnke & Geiger, (2012) instances. These improvements indicate that the proposed greedy actor–neural critic architecture scales effectively with increasing routing flexibility and problem dimensionality, enabling more efficient exploitation of machine alternatives than conventional optimization approaches. Overall, while GANC-PPO-RL maintains competitive performance on small instances, its performance gains become increasingly pronounced for large-scale flexible job shop problems, highlighting its suitability for complex and realistic manufacturing environments.

Building on the single-objective makespan results, Table 5 evaluates all algorithms under a weighted multi-objective formulation that simultaneously considers makespan and total energy consumption using three preference settings: makespan-focused, balanced, and energy-focused. These configurations allow the scheduling behaviour to transition from time-oriented production to energy-efficient operation, as well as to an explicit compromise between the two conflicting objectives.

The experimental results indicate that the proposed GANC-PPO-RL framework consistently outperforms the benchmark algorithms across all instance sizes. The performance advantage becomes particularly pronounced for large-scale scheduling problems, where the search space grows exponentially due to routing flexibility. This improvement can be attributed to two key characteristics of the proposed framework. First, the continuous heuristic weight learning provided by the PPO actor enables flexible prioritization of scheduling criteria depending on the shop floor state. Second, the greedy action selection mechanism efficiently translates these learned priorities into feasible job–machine assignments, thereby avoiding the combinatorial explosion associated with direct action selection.

In the makespan-focused configuration, GANC-PPO-RL achieves the lowest makespan values for every instance; for example, in large-scale instances (IDs 5 and 6), it reduces makespan by approximately 16–20% compared to RL-QL, over 50% compared to PSO, and more than 60% compared to GA, while maintaining competitive energy consumption.

In the balanced configuration, GANC-PPO-RL simultaneously achieves both lower makespan and lower energy consumption, yielding a more favourable compromise; for instance, for instance ID 5, it attains comparable makespan to RL-QL while reducing energy consumption by approximately 12–15%, and significantly outperforms PSO and GA in both objectives.

In the energy-focused configuration, the advantage of GANC-PPO-RL remains evident, as it achieves substantial reductions in energy consumption while limiting the associated increase in makespan. For large and very large instances, energy savings of 15–25% relative to RL-QL and over 30% relative to PSO and GA are observed, demonstrating that the proposed framework effectively prioritizes energy-efficient machine selections without severely degrading production completion time.

As the instance size increases, the performance gap between GANC-PPO-RL and the benchmark methods becomes more pronounced. This is attributed to the neural critic's ability to capture long-term scheduling consequences across both objectives, combined with the greedy actor's fast conversion of PPO-learned continuous heuristic weights into feasible job–machine assignments. In contrast, GA suffers from slow convergence and poor adaptability, PSO struggles to maintain a stable balance between makespan and energy in high-dimensional search spaces, and Q-learning-based RL is limited by its discrete state–action representation. Overall, the quantitative results in Table 5 clearly demonstrate that GANC-PPO-RL provides the most effective, scalable, and robust solution for multi-objective flexible job shop scheduling.

Table 5
Multi-Objective Results under Different Configurations

Flexible Instance ID	Makespan focused configuration (80% makespan and 20% energy consumption)								Balanced configuration (50% makespan and 50% energy consumption)								Energy focused configuration (20% makespan and 80% energy consumption)							
	Makespan, C_{max} (min)				Energy consumption, E (kW)				Makespan, C_{max} (min)				Energy consumption, E (kW)				Makespan, C_{max} (min)				Energy consumption, E (kW)			
	GANC-PPO-RL	RL-QL	PSO	GA	GANC-PPO-RL	RL-QL	PSO	GA	GANC-PPO-RL	RL-QL	PSO	GA	GANC-PPO-RL	RL-QL	PSO	GA	GANC-PPO-RL	RL-QL	PSO	GA	GANC-PPO-RL	RL-QL	PSO	GA
1	1443	1438	1455	1420	102.8	103.2	100.9	97.2	1608	1588	1642	1574	99.3	98.8	97.2	92.3	2135	2296	2174	2160	93.9	94.0	93.9	91.6
2	56	54	57	54	4.2	5.2	5.1	5.2	64	62	65	63	4.1	4.9	6.4	5.4	68	71	72	69	4.0	5.9	6.0	5.5
3	1009	1108	1098	1025	146.1	157.9	142.4	147.6	1082	1123	1155	1137	145.6	146.6	137.0	143.1	1106	1142	1160	1132	140.2	144.0	134.1	139.9
4	421	446	454	448	60.8	63.8	64.7	66.2	445	462	463	455	59.6	61.1	62.4	62.4	463	492	496	480	59.2	61.0	61.8	62.1
5	422	484	528	553	129.9	208.4	223.3	243.1	423	503	577	562	128.7	147.1	223.0	239.1	447	528	591	584	128.5	131.5	217.6	223.4
6	417	486	472	469	79.6	123.0	126.7	129.7	435	482	699	765	79.2	90.8	124.9	127.2	452	597	979	1316	79.1	82.7	121.3	121.7
7	263	298	355	334	43.9	44.7	45.5	45.4	267	322	363	353	43.6	44.0	44.4	44.9	271	369	382	381	43.4	43.7	43.8	44.3
8	414	458	463	468	154.9	156.8	159.0	159.6	417	502	493	503	153.8	154.2	156.0	157.5	430	541	561	533	151.0	151.9	148.9	153.8

Table 6
Performance metrics for the flexible instances under balanced configuration

Flexible Instance ID	Average machine utilization rate, $\overline{UR}_k$ (in percentage %)				Standard deviation of machine utilization, σ_U (in percentage %)				Average machine idle time, $\overline{IT}_k$ (in min)				Average job completion time, $\bar{C}_n$ (in min)			
	GANC-PPO-RL	RL-QL	PSO	GA	GANC-PPO-RL	RL-QL	PSO	GA	GANC-PPO-RL	RL-QL	PSO	GA	GANC-PPO-RL	RL-QL	PSO	GA
1	55.34	57.26	49.85	59.18	19.68	21.83	22.05	25.71	724.88	861.43	943.75	1209.62	957.17	1002.42	1038.64	945.30
2	39.06	38.10	36.20	37.40	26.25	26.96	28.46	27.68	39.02	41.85	44.29	43.12	41.80	43.21	45.18	42.70
3	70.68	67.90	63.42	65.11	9.85	12.69	15.99	14.56	317.20	358.47	392.73	378.55	879.67	905.44	938.86	920.17
4	68.49	66.82	65.10	67.44	14.59	15.46	17.25	15.98	146.53	158.28	165.86	152.64	343.93	356.22	361.89	349.71
5	26.54	24.81	15.44	15.18	29.84	31.28	35.32	34.11	310.73	372.45	428.64	409.80	241.11	298.66	335.26	321.29
6	24.33	22.02	16.65	15.59	26.30	29.14	34.88	37.62	329.17	621.57	749.32	909.63	239.36	502.18	645.47	783.80
7	59.60	56.20	52.45	53.86	12.46	13.84	16.94	15.74	107.88	138.40	159.74	151.63	187.87	214.36	238.52	229.47
8	63.56	60.46	57.84	58.22	12.55	13.34	15.64	15.27	151.97	184.60	198.17	199.34	284.49	316.84	328.55	329.41

In flexible job shop scheduling, overall schedule quality is not fully captured by makespan and energy alone, since routing flexibility strongly affects how evenly machines are loaded and how smoothly jobs flow through the system. Therefore, additional performance indicators average machine utilization rate, standard deviation of machine utilization, average machine idle time, and average job completion time are used to evaluate how effectively the schedule exploits machine alternatives, balances workload, and reduces waiting across the shop floor. These metrics provide a comprehensive assessment of operational efficiency and production stability in flexible manufacturing environments without dynamic disruptions.

Under the balanced multi-objective configuration in the flexible job shop scheduling problem without dynamic events, Table 6 compares the algorithms using four key performance metrics: average machine utilization rate, standard deviation of machine utilization, average machine idle time, and average job completion time. These metrics provide a deeper operational view of schedule quality beyond makespan and energy. Across all flexible job shop instances, the proposed GANC-PPO-RL achieves the highest utilization rates, lowest idle times, and shortest job completion times, indicating more efficient use of machines and smoother job flow through the system. The standard deviation of machine utilization is also consistently lower, showing that the greedy actor guided by the neural critic distributes workload more evenly and avoids bottlenecks, especially as the instance size increases.

In comparison, the Q-learning-based RL approach performs better than PSO and GA in most metrics due to its ability to learn adaptive scheduling policies, but its discrete state–action structure limits scalability and long-term optimization. PSO shows moderate performance but struggles with load balance and idle time as problem size grows, while GA exhibits the weakest utilization and longest completion times. The superior performance of GANC-PPO-RL is attributed to its continuous heuristic learning and greedy action selection, which together enable efficient machine usage, balanced workload distribution, and faster job completion in large-scale flexible job shop environments.

As shown in Table 7, the complete GANC-PPO-RL framework consistently outperforms its ablated variants in both makespan and energy consumption. Compared to the model without the greedy actor, the proposed framework achieves roughly a 2% reduction in makespan and about a 3–4% improvement in energy efficiency, indicating the importance of directed action selection. When the neural critic is removed, the performance gap remains evident, with nearly 1% improvement in makespan and around 2–3% lower energy usage, along with better stability. The simultaneous removal of both components results in the largest degradation, where the proposed approach demonstrates approximately 3–4% improvement in makespan and over 6% reduction in energy consumption. These results confirm that both components contribute synergistically to improving scheduling quality and operational efficiency.

Table 7
Ablation studies – Impact of greedy action and neural critic components on Flexible ID 8 of balanced configuration

Model Variant	Average makespan (min) Mean ± SD	Average energy consumption (kW) Mean ± SD
GANC-PPO-RL	426±14	155.8±1.8
without Greedy Actor	434±12	161.2±2.4
without Neural Critic	431±22	159.6±2.1
without Greedy Actor and Neural Critic	441±28	166.±3.2

5.2 Performance under Dynamic Disruptions

Table 8
Dynamic instances and their description

Dynamic Instance ID	Disruption description
1A-F	Low disruption: 0.5 machine breakdown per hour – Full breakdown without recovery
1A-R	Low disruption: 0.5 machine breakdown per hour – Breakdown with recovery after 30 min.
1A-P	Low disruption: 0.5 machine breakdown per hour – Partial breakdown (reduced performance).
1B-F	High disruption: 1.5 machine breakdown per hour – Full breakdown without recovery
1B-R	High disruption: 1.5 machine breakdown per hour – Breakdown with recovery after 30 min.
1B-P	High disruption: 1.5 machine breakdown per hour – Partial breakdown (reduced performance).
2A	Low disruption: 0.5 new job per hour arrival
2B	High disruption: 1.5 new job per hour arrival
3A	Low disruption: 0.5 Job cancellation per hour
3B	High disruption: 1.5 Job cancellation per hour
4A	Combined Low Disruption (1A-F +2A+3A)
4B	Combined High Disruption (1B-F +2B+3B)

To evaluate the proposed GANC-PPO-RL framework under real-time shop floor disturbances, dynamic events were simulated using a Poisson arrival process. As summarized in Table 8, three major types of disruptions, machine breakdowns, job cancellations, job insertions, and their combined occurrence were considered under high, medium, and low event intensity

levels, controlled through the event occurrence (EO) and event occurrence rate (EOR or λ) parameters. Higher EO and λ values represent highly volatile environments with frequent disruptions, while lower values correspond to relatively stable operating conditions. This controlled stochastic modeling enables a systematic evaluation of scheduling performance across different levels of uncertainty and allows the learning-based scheduler to be tested under realistic and scalable dynamic production scenarios. In a dynamic flexible job shop, the occurrence of machine breakdowns, job cancellations, and job insertions leads to continuous variations in makespan and total energy consumption, while the average flow time reflects how efficiently newly inserted jobs are absorbed into the ongoing production schedule. Since these disruptions simultaneously affect production completion, energy usage, and the responsiveness to new job arrivals, individual performance metrics alone do not adequately describe schedule stability. The Robustness Index (RI) is therefore used to measure how well the proposed scheduling policy preserves makespan, energy consumption, and flow time under stochastic dynamic conditions.

Table 9
Dynamic performance metrics for very large instance with balanced configuration using GANC-PPO-RL algorithm

Dynamic Instance ID	Makespan (min)	Makespan Robustness index, $RI_{C_{max}}$	Energy consumption (kW)	Energy consumption Robustness index, RI_E	Average job flow time, $\bar{F}$
No dynamic disruption	417	1	153.8	1	284
1A-F	419	1.004	157.3	1.022	297
1A-R	412	0.988	153.6	0.998	290
1A-P	412.59	0.989	155.8	1.01	297.56
1B-F	439	1.052	150.3	0.977	297.59
1B-R	415	0.995	150.3	0.998	292.56
1B-P	408.85	0.980	153.5	1.00004	294.32
2A	522.56	1.253	152.9	1.022	292.70
2B	563.83	1.352	157.3	1.091	287.51
3A	406	0.973	167.8	0.988	286.78
3B	409.6	0.9822	152.1	0.983	290.87
4A	433	1.0383	151.2	0.985	299.51
4B	412	0.988	155.2	1.008	293.82

The robustness of the proposed GANC-PPO-RL framework under stochastic dynamic disruptions is quantitatively evaluated using the robustness indices (RI) of makespan, total energy consumption, and average job flow time, as reported in Table 9 and Fig.7. The RI values are computed relative to the no-disruption baseline, where an RI value close to 1 indicates strong resilience and minimal performance degradation under dynamic events.

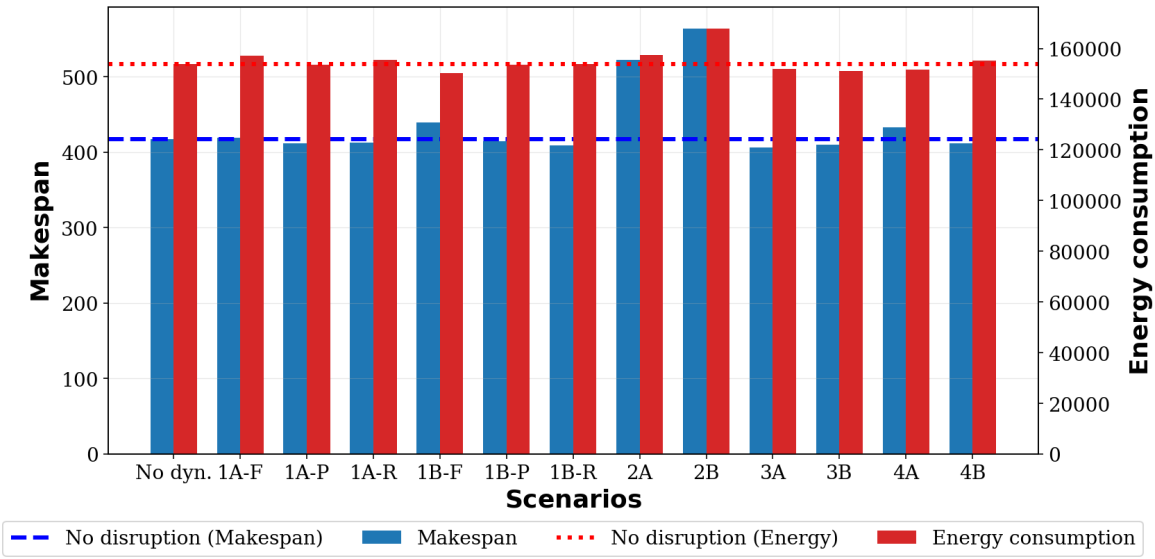

Fig. 7. Illustration of system robustness under various dynamic disruptions

Across all disruption scenarios, the makespan robustness index remains tightly bounded within a narrow range, with most values lying between 0.97 and 1.05, even under high-intensity machine breakdowns, job insertions, and combined disruptions. Similarly, the energy consumption robustness index consistently stays close to unity, typically within ±2–3% of the baseline, indicating that the scheduling policy preserves energy-efficient machine selection despite frequent stochastic disturbances. The average job flow time also exhibits limited variation, showing only moderate increases under high disruption levels, which demonstrates the framework's ability to rapidly absorb newly inserted jobs and reconfigure schedules without causing excessive waiting. The complete Gantt charts for both flexible and dynamic scenarios are provided in the supplementary Zenodo repository.

These results indicate that the learned greedy actor–neural critic policy maintains near-optimal scheduling behaviour under dynamic conditions, rather than reacting myopically to individual disruptions. The PPO-based policy effectively captures long-term system dynamics, allowing it to stabilize makespan and energy performance while preserving responsiveness to job arrivals and cancellations. Consequently, the consistently high robustness indices confirm that the proposed GANC-PPO-RL framework does not merely optimize static schedules, but learns disruption-tolerant scheduling strategies that are well suited for deployment in highly volatile and realistic flexible manufacturing environments.

6. Conclusion and future work

This study develops a Greedy Actor Neural Critic Proximal Policy Optimization reinforcement learning framework to solve the Dynamic Flexible Job Shop Scheduling Problem in complex manufacturing environments. The scheduling process is modelled as a Markov Decision Process that integrates policy learning with heuristic decision support. Unlike conventional reinforcement learning methods that search directly in a large job machine assignment space, the proposed model introduces a greedy actor and neural critic mechanism. The PPO policy network generates priority weights, while a deterministic greedy actor converts these priorities into feasible scheduling actions. This greedy actor reduces the complexity of the action space and improves exploration efficiency while maintaining high quality scheduling decisions.

Experiments on benchmark scheduling instances show consistent improvements in makespan and energy performance. The neural critic evaluates long term scheduling consequences and captures delayed system effects such as downstream congestion and machine utilization imbalance. The learning process therefore produces stable policies that outperform conventional reinforcement learning and heuristic approaches. Manufacturing environments are increasingly dynamic because of frequent job arrivals, machine disruptions, and changing production demands. The use of reinforcement learning methods such as Proximal Policy Optimization allows scheduling policies to adapt through continuous interaction with the shop floor environment.

From a managerial perspective, the proposed method supports real time production planning in smart factories. It can assist managers in improving machine utilization, reducing production delays, and controlling energy consumption. Future research may extend the model to incorporate additional objectives including due date performance, maintenance planning, and carbon emission reduction. Validation with industrial datasets and integration with digital twin systems would further strengthen its practical relevance.

Acknowledgement/Funding

The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Data availability statement

The benchmark datasets used in this study are publicly available at Zenodo

(DOI: https://doi.org/10.5281/zenodo.18865577).

The complete high-resolution Gantt charts corresponding to both flexible and dynamic scheduling scenarios are also accessible through a separate Zenodo repository
(DOI: https://doi.org/10.5281/zenodo.18865740).

Author contribution

Selva Kumar Chandrasekar: Conceptualization, Formal analysis, Investigation, Methodology, Project administration, Software, Supervision, Validation, Visualization, Writing - review.

Hariss Kumar Shanmugaprabu: Conceptualization, Formal analysis, Investigation, Validation, Writing - original draft

Aswath Mani: Data curation, Investigation, Methodology, Software, Validation, Writing - review.

Nishanth Sami Raja Murugan: Formal analysis, Investigation, Visualization, Writing - review

Conflict of interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Behnke, D., & Geiger, M. J. (2012). *Test instances for the flexible job shop scheduling problem with work centers* [Application/pdf]. https://doi.org/10.24405/436

Brandimarte, P. (1993). Routing and scheduling in a flexible job shop by tabu search. *Annals of Operations Research*, *41*(3), 157–183. https://doi.org/10.1007/BF02023073

Chandrasekar, S. K., Sasikumar, V. K., & Rajkumar, S. K. (2025). Flexible job shop scheduling using Jaya-Tabu search algorithm. *International Journal of Systems Science: Operations & Logistics*, *12*(1), 2573992. https://doi.org/10.1080/23302674.2025.2573992

Chang, H.-C., Chen, Y.-P., Liu, T.-K., & Chou, J.-H. (2015). Solving the Flexible Job Shop Scheduling Problem With Makespan Optimization by Using a Hybrid Taguchi-Genetic Algorithm. *IEEE Access*, *3*, 1740–1754. https://doi.org/10.1109/ACCESS.2015.2481463

Chang, J., Yu, D., Hu, Y., He, W., & Yu, H. (2022). Deep Reinforcement Learning for Dynamic Flexible Job Shop Scheduling with Random Job Arrival. *Processes*, *10*(4), 760. https://doi.org/10.3390/pr10040760

Chaudhry, I. A., & Khan, A. A. (2016). A research survey: Review of flexible job shop scheduling techniques. *International Transactions in Operational Research*, *23*(3), 551–591. https://doi.org/10.1111/itor.12199

Chung, K. T., Lee, C. K. M., & Tsang, Y. P. (2025). Neural combinatorial optimization with reinforcement learning in industrial engineering: A survey. *Artificial Intelligence Review*, *58*(5), 130. https://doi.org/10.1007/s10462-024-11045-1

Dauzère-Pérès, S., Ding, J., Shen, L., & Tamssaouet, K. (2024). The flexible job shop scheduling problem: A review. *European Journal of Operational Research*, *314*(2), 409–432. https://doi.org/10.1016/j.ejor.2023.05.017

Destouet, C., Tlahig, H., Bettayeb, B., & Mazari, B. (2025). Systematic review and future directions in dynamic flexible job shop scheduling: A decade of research. *Journal of Intelligent Manufacturing*. https://doi.org/10.1007/s10845-025-02645-x

Du, Y., Li, J., Li, C., & Duan, P. (2024). A Reinforcement Learning Approach for Flexible Job Shop Scheduling Problem With Crane Transportation and Setup Times. *IEEE Transactions on Neural Networks and Learning Systems*, *35*(4), 5695–5709. https://doi.org/10.1109/TNNLS.2022.3208942

Fattahi, P., Saidi Mehrabad, M., & Jolai, F. (2007a). Mathematical modeling and heuristic approaches to flexible job shop scheduling problems. *Journal of Intelligent Manufacturing*, *18*(3), 331–342. https://doi.org/10.1007/s10845-007-0026-8

Fattahi, P., Saidi Mehrabad, M., & Jolai, F. (2007b). Mathematical modeling and heuristic approaches to flexible job shop scheduling problems. *Journal of Intelligent Manufacturing*, *18*(3), 331–342. https://doi.org/10.1007/s10845-007-0026-8

Han, B. A., & Yang, J. J. (2021). A Deep Reinforcement Learning Based Solution for Flexible Job Shop Scheduling Problem. *International Journal of Simulation Modelling*, *20*(2), 375–386. https://doi.org/10.2507/IJSIMM20-2-CO7

Hurink, J., Jurisch, B., & Thole, M. (1994). Tabu search for the job-shop scheduling problem with multi-purpose machines. *OR Spektrum*, *15*(4), 205–215. https://doi.org/10.1007/BF01719451

Lei, K., Guo, P., Zhao, W., Wang, Y., Qian, L., Meng, X., & Tang, L. (2022). A multi-action deep reinforcement learning framework for flexible Job-shop scheduling problem. *Expert Systems with Applications*, *205*, 117796. https://doi.org/10.1016/j.eswa.2022.117796

Li, X., Guo, X., Tang, H., Wu, R., Wang, L., Pang, S., Liu, Z., Xu, W., & Li, X. (2022). Survey of integrated flexible job shop scheduling problems. *Computers & Industrial Engineering*, *174*, 108786. https://doi.org/10.1016/j.cie.2022.108786

Liang, X., Ma, Y., Feng, Y., & Liu, Z. (2021). *PTR-PPO: Proximal Policy Optimization with Prioritized Trajectory Replay* (Version 2). arXiv. https://doi.org/10.48550/ARXIV.2112.03798

Lopes Silva, M. A., De Souza, S. R., Freitas Souza, M. J., & Bazzan, A. L. C. (2019). A reinforcement learning-based multi-agent framework applied for solving routing and scheduling problems. *Expert Systems with Applications*, *131*, 148–171. https://doi.org/10.1016/j.eswa.2019.04.056

Park, M.-J., & Ham, A. (2022). Energy-aware flexible job shop scheduling under time-of-use pricing. *International Journal of Production Economics*, *248*, 108507. https://doi.org/10.1016/j.ijpe.2022.108507

Peng, N., Zheng, Y., Xiao, Z., Gong, G., Huang, D., Liu, X., Zhu, K., & Luo, Q. (2025). Multi-objective dynamic distributed flexible job shop scheduling problem considering uncertain processing time. *Cluster Computing*, *28*(3), 185. https://doi.org/10.1007/s10586-024-04803-x

Rahmani, D., & Ramezanian, R. (2016). A stable reactive approach in dynamic flexible flow shop scheduling with unexpected disruptions: A case study. *Computers & Industrial Engineering*, *98*, 360–372. https://doi.org/10.1016/j.cie.2016.06.018

Rajabinasab, A., & Mansour, S. (2011). Dynamic flexible job shop scheduling with alternative process plans: An agent-based approach. *The International Journal of Advanced Manufacturing Technology*, *54*(9–12), 1091–1107. https://doi.org/10.1007/s00170-010-2986-7

Salido, M. A., Escamilla, J., Giret, A., & Barber, F. (2016). A genetic algorithm for energy-efficiency in job-shop scheduling. *The International Journal of Advanced Manufacturing Technology*, *85*(5–8), 1303–1314. https://doi.org/10.1007/s00170-015-7987-0

Saouabi, M. D. E., Nouri, H. E., & Belkahla Driss, O. (2024). A two-level evolutionary algorithm for dynamic scheduling in flexible job shop environment. *Evolutionary Intelligence*, *17*(5–6), 4133–4153. https://doi.org/10.1007/s12065-024-00976-x

Shahgholi Zadeh, M., Katebi, Y., & Doniavi, A. (2019). A heuristic model for dynamic flexible job shop scheduling problem considering variable processing times. *International Journal of Production Research*, *57*(10), 3020–3035. https://doi.org/10.1080/00207543.2018.1524165

Shen, X.-N., Han, Y., & Fu, J.-Z. (2017). Robustness measures and robust scheduling for multi-objective stochastic flexible job shop scheduling problems. *Soft Computing*, *21*(21), 6531–6554. https://doi.org/10.1007/s00500-016-2245-4

Sitahong, A., Wang, G., Yuan, Y., Wubuli, A., Mo, P., & Chen, Y. (2025). Energy-Efficient Scheduling in Dynamic Flexible Job Shops: A Review. *Sustainability*, *17*(24), 11009. https://doi.org/10.3390/su172411009

Sun, K., Zheng, D., Song, H., Cheng, Z., Lang, X., Yuan, W., & Wang, J. (2023). Hybrid genetic algorithm with variable neighborhood search for flexible job shop scheduling problem in a machining system. *Expert Systems with Applications*, *215*, 119359. https://doi.org/10.1016/j.eswa.2022.119359

Tang, D., Dai, M., Salido, M. A., & Giret, A. (2016). Energy-efficient dynamic scheduling for a flexible flow shop using an improved particle swarm optimization. *Computers in Industry*, *81*, 82–95. https://doi.org/10.1016/j.compind.2015.10.001

Vital-Soto, A., Baki, M. F., & Azab, A. (2023). A multi-objective mathematical model and evolutionary algorithm for the dual-resource flexible job-shop scheduling problem with sequencing flexibility. *Flexible Services and Manufacturing Journal*, *35*(3), 626–668. https://doi.org/10.1007/s10696-022-09446-x

Wang, L., Pan, Z., & Wang, J. (2021). A Review of Reinforcement Learning Based Intelligent Optimization for Manufacturing Scheduling. *Complex System Modeling and Simulation*, *1*(4), 257–270. https://doi.org/10.23919/CSMS.2021.0027

Yu, W., Zhang, L., & Ge, N. (2022). An adaptive multiobjective evolutionary algorithm for dynamic multiobjective flexible scheduling problem. *International Journal of Intelligent Systems*, *37*(12), 12335–12366. https://doi.org/10.1002/int.23090

Zhang, M., Tao, F., & Nee, A. Y. C. (2021). Digital Twin Enhanced Dynamic Job-Shop Scheduling. *Journal of Manufacturing Systems*, *58*, 146–156. https://doi.org/10.1016/j.jmsy.2020.04.008

Zhang, W., Zhao, F., Feng, B., & Mei, X. (2025). A Reinforcement Learning Control Framework Based on Scalable Graph Transformer for Large-Scale Fuzzy Job Shop Scheduling Problems. *IEEE Transactions on Neural Networks and Learning Systems*, *36*(9), 16521–16533. https://doi.org/10.1109/TNNLS.2025.3569868

Zhang, X., & Zhu, G.-Y. (2025). A literature review of reinforcement learning methods applied to job-shop scheduling problems. *Computers & Operations Research*, *175*, 106929. https://doi.org/10.1016/j.cor.2024.106929

Zhang, Y., Zhu, H., & Tang, D. (2019). An improved hybrid particle swarm optimization for multi-objective flexible job-shop scheduling problem. *Kybernetes*, *49*(12), 2873–2892. https://doi.org/10.1108/K-06-2019-0430

Zhao, C., & Deng, N. (2023). An actor-critic framework based on deep reinforcement learning for addressing flexible job shop scheduling problems. *Mathematical Biosciences and Engineering*, *21*(1), 1445–1471. https://doi.org/10.3934/mbe.2024062

Zhu, J., Li, X., Ruiz, R., Li, W., Huang, H., & Zomaya, A. Y. (2020). Scheduling Periodical Multi-Stage Jobs With Fuzziness to Elastic Cloud Resources. *IEEE Transactions on Parallel and Distributed Systems*, *31*(12), 2819–2833. https://doi.org/10.1109/TPDS.2020.3004134

© 2026 by the authors; licensee Growing Science, Canada. This is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC-BY) license (http://creativecommons.org/licenses/by/4.0/).