Generative AI-Driven Optimization in Flexible and Reconfigurable Manufacturing Systems

Salah Hammedi
Networked Objects, Control, and Communication Systems (NOCCS), ENISo, University of Sousse, Tunisia | Electrical Engineering Department, National School of Engineers of Monastir, Monastir University, Tunisia
salahhammedi@yahoo.com

Hicham Chaoui
Department of Electronics, Carleton University, Ottawa, Canada | Electrical and Computer Engineering Department, Old Dominion University, Norfolk, USA
Hicham.Chaoui@carleton.ca (corresponding author)

Lotfi Nabli
Networked Objects, Control, and Communication Systems (NOCCS), ENISo, University of Sousse, Tunisia | Electrical Engineering Department, National School of Engineers of Monastir, Monastir University, Tunisia
lotfi.nabli@enim-u-monastir.tn

Received: 6 November 2025 | Revised: 22 December 2025 | Accepted: 3 January 2026

 DOI: https://doi.org/10.48084/etasr.16077

ABSTRACT

Flexible and Reconfigurable Manufacturing Systems (FRMSs) are essential for coping with variability in modern production environments; however, efficient scheduling and rapid reconfiguration remain challenging. This paper presents a hybrid optimization framework that integrates Colored Petri Net (CPN) modeling with Generative Artificial Intelligence (GenAI) to enhance scheduling performance and system adaptability. The CPN formalism ensures verifiable modeling of system dynamics, while a transformer-based generative model produces candidate scheduling and reconfiguration strategies. Simulation experiments were conducted under static, dynamic, and adaptive scenarios, including machine breakdowns and dynamic job arrivals. Performance was evaluated using makespan, mean flow time, machine utilization, and reconfiguration latency. The results indicate that the proposed approach reduces the makespan by approximately 11–12% and improves machine utilization by 7–9% compared to classical heuristics and genetic algorithms, while in dynamic and adaptive scenarios, reconfiguration latency is reduced by up to 26%. These findings demonstrate that combining formal Petri net models with GenAI provides an effective mathematical framework for adaptive optimization in FRMSs.

Keywords-Flexible and Reconfigurable Manufacturing Systems (FRMSs); generative AI; Petri nets; intelligent scheduling; performance optimization; Industry 4.0

I. INTRODUCTION

The global manufacturing sector is undergoing a transformative shift under the banner of Industry 4.0, where the demands for flexibility, adaptability, and intelligent automation are driving innovation at unprecedented levels [1, 2]. Flexible Manufacturing Systems (FMS) and Reconfigurable Manufacturing Systems (RMS) have emerged as critical enablers, offering modularity and scalable responses to rapidly changing market requirements [3, 4]. Despite these advances, significant challenges persist in efficient order scheduling, resource allocation, and dynamic reconfiguration, especially under disturbances such as machine breakdowns or fluctuating demand [5, 6]. In recent years, the rise of Generative Artificial Intelligence (GenAI), including transformer-based models, diffusion architectures, and Large Language Models (LLMs), has opened new possibilities in manufacturing. Such systems can generate novel schedules, simulate reconfiguration pathways, and predict operational bottlenecks in real time [7, 8]. For example, a 2025 review [9] highlights how GenAI can augment adaptive control in dynamic manufacturing environments. Innovations like Industry 6.0, which couple GenAI with swarms of heterogeneous robots to fully automate design and production workflows, demonstrate a dramatic leap in autonomy and efficiency [10].

Industry leaders are already piloting GenAI solutions: BMW's Factory Genius, an internal LLM-powered digital assistant, accelerates maintenance inquiries and technical problem-solving on the shop floor [11]. Their AIonic multi-agent platform further embeds GenAI across production, procurement, and quality control to elevate operational efficiency [12]. These initiatives exemplify the industrial scalability and practical utility of this technology.

On the modeling front, Petri nets remain a robust formalism for representing discrete-event manufacturing systems. Their strengths in modeling concurrency, synchronization, and resource sharing make them ideal for formal analysis and control [13]. However, challenges in scaling Petri net models to encompass real-world complexity, including deadlock prevention and reconfiguration logic, persist [14].

Against this backdrop, the proposed approach integrates Petri net-based system modeling with GenAI-generated scheduling and reconfiguration strategies. This hybrid framework enables the generation of alternative operational plans, their simulation via the Petri net model, and dynamic selection of optimal strategies under uncertainty. The novelty lies in the fusion of formal modeling with generative decision-making to deliver robust, adaptive, and explainable manufacturing control.

Manufacturing research has progressively shifted from classical optimization-based techniques toward AI-driven and generative approaches to address the challenges of flexibility, adaptability, and dynamic reconfiguration. This section reviews representative studies from 2019 to 2025 in four main research directions: (i) classical optimization, (ii) AI-based scheduling, (iii) Petri net-AI hybrids, and (iv) emerging generative models. Table I presents a comparative summary of the most relevant works, highlighting their methods, evaluation metrics, strengths, and limitations.

A. Classical Optimization for FMS/RMS

Early studies on FMS/RMS mainly employed mathematical and heuristic optimization techniques such as Mixed-Integer Linear Programming (MILP), dispatching rules (SPT/EDD), and meta-heuristics such as Genetic Algorithms (GA), Tabu Search, and Simulated Annealing. Although exact optimization guarantees optimal solutions, its scalability quickly deteriorates when the size and reconfiguration events increase. Meta-heuristics offer improved tractability and near-optimal performance but often require manual parameter tuning and lack adaptability to disturbances. Recent works have attempted to combine formal modeling (e.g., Petri nets) with search heuristics to better represent resource sharing and concurrency in RMS environments [15-17]. However, such hybrids still face limitations in dynamic, disturbance-rich contexts [18, 19].

B. AI Techniques for Scheduling (DRL, GNNs, MARL, XAI)

With the advancement of Deep Reinforcement Learning (DRL), scheduling problems have witnessed a shift from handcrafted rules toward learned dispatching policies. DRL agents trained across multiple scenarios can generalize to unseen conditions, outperforming static heuristics in dynamic job-shop scheduling [20, 21].

Graph Neural Networks (GNNs) enhance learning by representing operations and machines as structured graphs, leading to better scalability [22]. Multi-Agent Reinforcement Learning (MARL) also shows potential for decentralized decision-making in flexible production systems [23]. In addition, Explainable AI (XAI) methods have been introduced to interpret scheduling decisions, increasing confidence in AI-based planning [24].

C. Petri Net-AI Hybrids for RMS/FMS

Hybrid approaches combining Petri nets with AI algorithms, such as GA, DRL, or fuzzy optimization, are increasingly popular for modeling, monitoring, and optimizing RMS behavior [15-17, 25]. In these frameworks, Petri nets capture discrete events, resource conflicts, and process dependencies, while the AI component proposes optimized schedules or reconfiguration actions. These hybrid models have been successfully applied to intralogistics, AGV routing, and dynamic resource scheduling, offering improved throughput and reconfiguration adaptability [25]. Nevertheless, high simulation complexity and limited real-time applicability remain open challenges.

D. Generative Models and LLMs in Manufacturing Optimization

Recent developments in Generative AI (GenAI), including GANs, diffusion models, and LLMs, have opened new perspectives for industrial optimization and decision support. Generative models are now used for scenario generation, synthetic data augmentation, and candidate plan creation, significantly reducing computational effort and improving robustness under rare events [26]. Meanwhile, LLMs have been integrated as planning and orchestration layers capable of combining reasoning, simulation control, and human-in-loop decision-making [27-29].

Machine-learning-based decision support has begun to transform scheduling on heterogeneous machines, as evidenced by light-gradient boosting implementations for non-identical parallel machines [30]. Other works examine discrete-event simulation methods to improve resource utilization and scheduling robustness in semi-automated production contexts [31]. These studies reinforce the industry shift toward data-driven, adaptable scheduling frameworks, but even so, they tend to focus on predictive or heuristic machine-learning approaches rather than generative planning or formal reconfigurable system modeling. In contrast, the proposed framework unites formal discrete-event modeling via colored Petri nets with generative AI-driven schedule synthesis, thereby addressing the scheduling-plus-reconfiguration challenge in highly flexible manufacturing systems.

As summarized in Table I, most current approaches still lack formal integration with discrete-event simulation frameworks, which limits their verification and industrial deployment [32]. Substantial progress has been made in integrating optimization, AI, and Petri-net frameworks for intelligent scheduling in reconfigurable manufacturing systems. However, end-to-end generative pipelines that jointly leverage formal modeling, generative reasoning, and discrete-event simulation remain scarce.

TABLE I. COMPARATIVE SUMMARY OF SELECTED STUDIES (2019–2025)

Ref.	Configuration/ method	Metrics evaluated	Strengths	Limitations
[15]	Petri nets + GA in RMS	Makespan, resource utilization	Captures concurrency; improved throughput	Limited adaptability under disturbances
[20]	DRL agents for job-shop scheduling	Reward, makespan, tardiness	Robust generalization to variable demand	High training cost; limited interpretability
[25]	Petri nets + dynamic optimization for AGV routing	Throughput, routing costs	Good discrete-event modeling; state observability	High computational overhead
[26]	GAN/diffusion models for plan generation	Synthetic data quality, plan diversity	Fast scenario generation; covers rare events	No benchmark for real execution feasibility
[27, 28]	LLMs + solver/simulator APIs	Planning time, human workload	Enables fast drafting, human-in-loop design	Missing formal validation layer

This study addresses this gap by proposing an integrated framework that unites Petri-net-based system modeling, GenAI-driven plan generation, and simulation-based validation to achieve adaptive, explainable, and verifiable scheduling in RMS environments.

II. PROPOSED METHODOLOGY

The proposed method introduces a hybrid framework that integrates Colored Petri Nets (CPN) with GenAI to achieve intelligent, adaptive, and optimized scheduling in Flexible and Reconfigurable Manufacturing Systems (FRMSs). This approach combines the formal modeling precision of Petri nets with the creative generative capabilities of transformer-based AI models to dynamically produce feasible and high-performance schedules. This section details the architecture, modeling principles, optimization algorithm, and validation through a real-case simulation study, illustrating how GenAI enhances system responsiveness and decision-making in reconfigurable production environments.

A. Overview of the Hybrid FRMS–GenAI Framework

The proposed method presents an integrated framework that combines formal discrete-event system modeling with GenAI to achieve adaptive, efficient, and intelligent scheduling in FRMSs. This integration aims to overcome the limitations of traditional scheduling approaches by enabling the system to autonomously generate, evaluate, and adapt production plans under changing operational conditions such as demand fluctuations, machine breakdowns, or dynamic reconfiguration events. The framework is designed to ensure both analytical rigor through formal modeling and creative adaptability through data-driven generation, resulting in a decision-support mechanism that is robust, scalable, and explainable.

As illustrated in Figure 1, the proposed architecture is structured into four interconnected modules, each fulfilling a specific role in the end-to-end optimization process:

1. System Modeling Layer: This foundational layer employs CPNs to model the structural and behavioral dynamics of the manufacturing system. Machines, buffers, and conveyors are represented as places, while operations such as machining, transport, or setup changes are represented as transitions. Tokens encode job characteristics such as part type, processing time, and routing flexibility, allowing the system to simulate multiple product families and reconfiguration states. This layer provides a digital twin of the production environment, capturing concurrency, synchronization, and resource contention in real time. It ensures that the generated schedules remain logically feasible and formally verifiable through Petri-net-based properties such as liveness, reachability, and boundedness.
2. Data Exchange Layer: The second module serves as an interface between simulation and AI generation. Event logs and token trajectories from the Petri Net simulation are transformed into structured datasets containing features such as machine utilization, queue lengths, job completion ratios, and transition firing frequencies. These sequences form the input representation for the GenAI model, effectively encoding the evolving production context. The continuous feedback from the CPN model enables the generative model to adapt its scheduling outputs dynamically, ensuring responsiveness to real-time events.
3. Generative Optimization Layer: In this core module, a transformer-based GenAI model is trained to produce alternative scheduling and reconfiguration plans. Unlike deterministic optimization or static learning methods, this generative model explores a broad solution space by producing diverse candidate schedules that comply with production constraints while optimizing objectives such as makespan, tardiness, and throughput. Each generated schedule reflects both learned historical patterns and current CPN-derived states, allowing the model to propose innovative yet realistic configurations. By leveraging attention mechanisms, the model captures long-term dependencies among jobs and resources, making it particularly effective in complex reconfigurable environments where sequence decisions propagate through multiple stages.
4. Evaluation and Decision Layer: The final layer performs simulation-based evaluation of all candidate schedules produced by the GenAI model. Each plan is validated within the CPN simulation environment to ensure compliance with operational constraints and to compute quantitative performance indicators (e.g., average completion time, resource utilization, and reconfiguration cost). A multi-objective decision module then ranks and selects the most suitable plan according to Pareto-optimality principles. This real-time decision-making capability enables the FRMS to switch adaptively between feasible configurations without halting production, thereby enhancing system resilience and responsiveness.

Overall, the hybrid FRMS-GenAI framework establishes a continuous feedback and learning loop between the formal model and the generative intelligence. The Petri Net provides structural consistency and constraint verification, while the GenAI model contributes creativity, adaptability, and rapid re-optimization. Together, they form a self-improving scheduling ecosystem that evolves with operational data and external disturbances. Figure 1 illustrates the workflow of the proposed hybrid architecture, highlighting the interaction between the CPN model, the generative AI scheduler, the simulation-based evaluation loop, and the adaptive decision mechanism for continual optimization.

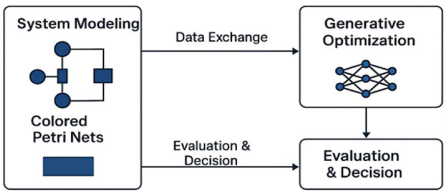

Fig. 1. Hybrid FRMS-GenAI framework.

B. System Modeling Using Colored Petri Nets

The proposed framework employs CPNs as a formal modeling technique to accurately represent the dynamic and reconfigurable behavior of an FRMS. CPNs extend classical Petri nets by introducing data types, or colors, enabling each token to carry multiple attributes such as job type, processing time, and routing path. This extension makes it possible to represent multi-product, multi-machine environments within a single integrated model, capturing both concurrent operations and system reconfigurations with high fidelity. In the constructed model:

- Places represent manufacturing states, such as machine availability, buffer capacities, and job queues.
- Transitions denote discrete events, including task initiation, machine operation, inspection, or reconfiguration.
- Arcs define logical and causal relationships between states and actions.
- Colored tokens encode dynamic job parameters, allowing simulation of multiple production scenarios simultaneously.

Formally, the CPN model for the FRMS can be defined as:

$$CPN = (P, T, A, \Sigma, C, N, E, G, I) \quad (1)$$

where P is the set of places, T is the set of transitions, $A \subseteq (P \times T) \cup (T \times P)$ represents arcs, Σ is the set of color types, C assigns color sets to places, N defines the node function, E specifies arc expressions, G is the guard function controlling transition firings, and I represents the initial marking, corresponding to the system's initial state.

The model was developed and simulated using CPN Tools (version 4.0), which provides formal verification capabilities to analyze liveness, boundedness, and reachability properties. These analyses ensure that the FRMS model operates without deadlocks, avoids resource conflicts, and maintains a consistent production flow even under dynamic reconfigurations.

1) Reconfigurability Modeling

To explicitly capture the reconfigurable nature of FRMSs, dedicated reconfiguration transitions are incorporated into the CPN structure. These transitions are conditionally enabled based on real-time operational conditions, such as machine overload, equipment failure, or routing deviations, thereby allowing the system to dynamically adapt its configuration during execution.

Reconfiguration decisions are governed by guard functions associated with specific transitions. A representative guard function for machine-level reconfiguration is defined as follows:

$$G_i(t) = \{1,\ if(load(M_i, t) > \theta_i) \vee\ (failure(M_i, t) = 1)\}; \{0, otherwise\} \quad (2)$$

where $G_i(t)$ is the binary guard function associated with machine M_i at time t, $load(M_i, t)$ denotes the instantaneous workload or utilization level of machine M_i, θ_i represents the predefined overload threshold for machine M_i, $failure(M_i, t) \in \{0,1\}$ is a failure indicator, with 1 denoting machine failure and 0 indicating normal operation, and $\vee$ denotes the logical OR operator.

When $G_i(t) = 1$, the corresponding reconfiguration transition becomes enabled in the Petri Net, triggering actions such as machine reassignment, alternative routing selection, or buffer redistribution. Conversely, when $G_i(t) = 0$, the system continues operating under its current configuration.

By formalizing reconfiguration conditions through explicitly defined guard functions, the proposed model ensures consistent, interpretable, and real-time adaptive behavior. This mechanism enables the FRMS to maintain production continuity and operational robustness under dynamic manufacturing conditions, including disturbances and structural changes.

2) Dual-Level Representation of the FRMS

To combine conceptual clarity with formal precision, two complementary representations of the FRMS were developed, as illustrated in Figures 2 and 3.

Figure 2 presents the conceptual CPN representation of the FRMS. It illustrates the flow of jobs across machines (M1–M3), buffers (B1–B2), and an inspection station, highlighting concurrency, synchronization, and reconfiguration capabilities. Colored tokens represent distinct product types, while transitions denote processing operations or configuration changes. This abstraction improves readability by providing a high-level architectural view of the system.

Figure 3 shows the formal CPN model implemented in CPN Tools v4.0, which provides an executable and verifiable representation of the system. Places represent machine, buffer, and inspection states, transitions model processing and reconfiguration actions, and arcs define causal dependencies. Token colors encode structured job attributes such as processing time, machine assignment, routing identifier, and priority level, enabling accurate simulation of heterogeneous and concurrent production flows.

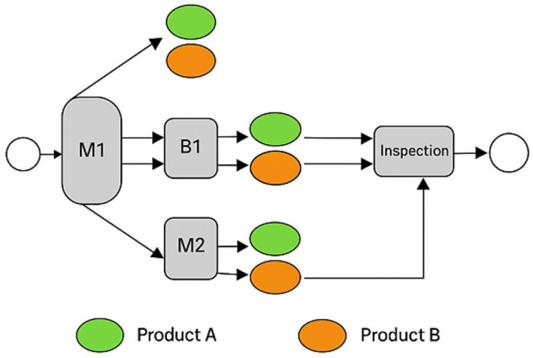

Fig. 2. Conceptual CPN of FRMS.

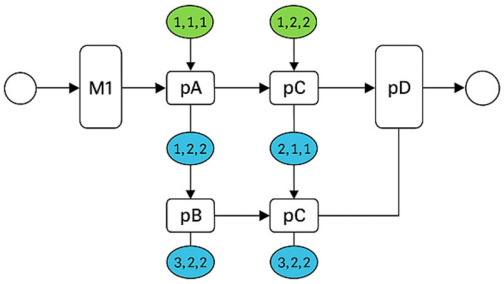

Fig. 3. Formal CPN model implemented in CPN tools.

a) CPN State Vectorization for GenAI Integration

To interface the formal CPN model with the transformer-based GenAI scheduler, the global marking of the Petri net is transformed into a structured numerical sequence at each decision epoch. This transformation ensures that symbolic discrete-event information is preserved while remaining compatible with deep learning architectures.

The encoding process follows three principles:

1. Completeness: all relevant system states (jobs, resources, buffers) are captured;
2. Consistency: identical CPN markings yield identical input sequences;
3. Feasibility preservation: precedence and resource constraints are respected through masking.

Each CPN marking is encoded into a fixed-length state vector composed of:

- Job-level features: product type (one-hot), remaining processing time (normalized), due-date urgency, current operation index;
- Resource-level features: machine availability (binary), utilization rate, reconfiguration state;
- Buffer-level features: queue length and waiting time statistics;
- Routing features: current and next machine identifiers (embedded categorical values).

The resulting state vector is ordered and segmented to form an input token sequence for the transformer model.

b) Algorithm 1: CPN-to-Sequence Encoding for Transformer Input

Algorithm 1 formalizes the transformation of a CPN marking into a transformer-compatible input sequence.

Algorithm 1: CPN-to-Sequence Encoding
Input:
  Colored Petri net model $M = (P, T, C, M_t)$
  Current marking M_t at decision time t
Output:
  Encoded state sequence X_t
Procedure:
1. Token Feature Extraction: For each token $\tau \in M_t$, extract job attributes $(j_\tau, p_\tau, o_\tau, d_\tau)$ where j_τ is job type, p_τ is the remaining processing time, o_τ is the current operation index, and d_τ is due-date urgency.
2. Resource State Extraction: For each machine M_i, extract: availability a_i, utilization u_i, and reconfiguration state r_i.
3. Buffer State Extraction: For each buffer B_k, extract queue length q_k and mean waiting time w_k.
4. Feature Encoding: Normalize continuous features; apply one-hot encoding for job types and embeddings for categorical identifiers (machines and routes).
5. Sequence Construction: Concatenate all encoded features into an ordered input sequence: $X_t = [x_1, x_2, \ldots, x_n]$
6. Constraint Masking: Apply a feasibility mask to exclude unavailable machines and precedence-violating operations.

Output: Return X_t as the transformer input representation.

By combining conceptual and formal CPN representations with an explicit state-to-sequence encoding mechanism, the proposed framework ensures full reproducibility of the GenAI scheduling layer. The conceptual model (Figure 2) supports system understanding, the technical model (Figure 3) enables accurate simulation and verification, and Algorithm 1 provides a precise bridge between discrete-event modeling and transformer-based generative learning. This integration forms the foundation for intelligent, adaptive scheduling and reconfiguration in complex FRMS environments.

C. Generative AI-Based Scheduling and Reconfiguration

To enhance decision-making in dynamic production environments, the second layer integrates GenAI with the CPN model developed. This integration enables the automatic generation and continuous adaptation of scheduling and reconfiguration decisions under uncertainty, variability, and structural changes in the manufacturing system.

Unlike conventional optimization or rule-based approaches, the generative layer employs transformer-based deep learning architectures capable of modeling long-range dependencies between jobs, machines, buffers, and reconfiguration states. This allows the framework to explore the scheduling solution space in a diverse yet constraint-aware manner, generating multiple feasible schedules that evolve in real time as system conditions change.

1) Framework Overview

Figure 4 shows the overall workflow of the GenAI+CPN framework, illustrating the closed-loop interaction between simulation, learning, and feedback. The CPN simulation environment provides the system state vector, capturing machine utilization, queue lengths, job priorities, and ongoing reconfiguration states. This encoded state serves as the context input (or "prompt") for the transformer-based GenAI model, which then produces candidate scheduling sequences tailored to current conditions. Each generated schedule is simulated within the CPN model to evaluate performance indicators such as makespan, tardiness, throughput, and load balancing.

The evaluation results are returned as reinforcement signals, establishing a cyclic feedback loop between symbolic modeling (CPN) and generative modeling (GenAI), ensuring continuous learning and adaptive optimization.

Figure 4 illustrates the interaction between the four key components, namely State Encoding, Generative Model, Simulation Evaluation, and Reinforcement Feedback, that together enable self-adaptive scheduling and reconfiguration.

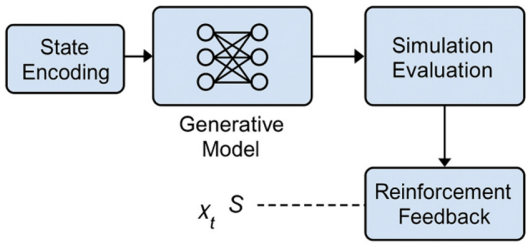

Fig. 4. Architecture of the GenAI and CPN integration loop.

2) Step-by-Step Method

The complete GenAI–CPN scheduling and reconfiguration procedure is summarized below:

Step 1—System Modeling: The RMS is formally modeled using CPNs in CPN Tools v4.0, where machines, buffers, robots, and reconfiguration options are represented by places and transitions.

Step 2—State Extraction and Encoding: At each scheduling decision point, the CPN model exports the current system state X_t, encoded as a numerical vector capturing:

- Job attributes (type, processing time, due date),
- Machine states (idle, busy, failed),
- Buffer occupancy levels,
- Routing and reconfiguration indicators.

Step 3—Generative Schedule Generation: The encoded state vector X_t is fed into a transformer-based generative model, which outputs a set of feasible scheduling sequences $S = \{S_1, S_2, \dots, S_n\}$. Each sequence satisfies resource availability and technological precedence constraints.

Step 4—Simulation-Based Evaluation: Each candidate schedule is executed within the CPN model to compute performance metrics such as:

- Makespan;
- Mean flow time;
- Machine utilization;
- Reconfiguration latency;

Step 5—Reinforcement Feedback and Policy Update: Based on simulation results, each schedule receives a reward signal and contributes to updating the model parameters.

Step 6—Schedule and Reconfiguration Deployment: The highest-performing schedule and associated reconfiguration actions are selected and applied to the RMS.

This loop is continuously repeated, enabling real-time adaptation.

3) GenAI Model Architecture and Implementation Details

- Model type: Transformer encoder–decoder
- Number of layers: 6 encoder layers, 6 decoder layers
- Hidden dimension: 512
- Attention heads: 8
- Activation function: GELU
- Optimizer: AdamW
- Learning rate: 1×10^{-4}
- Batch size: 64

The model was implemented in Python 3.11 using TensorFlow 2.15 and PyTorch 2.2.

4) Training Phase

The training phase aims to model the complex dependencies between dynamic manufacturing states and their corresponding optimal scheduling outcomes. Training data are extracted from extensive CPN simulations and historical production records, structured into (state, schedule) pairs.

- Input Representation: Each input vector encodes a system snapshot including job types, processing times, due dates, machine utilization, buffer occupancy, and global load distribution.
- Output Representation: Outputs correspond to near-optimal schedules identified via heuristic baselines or prior optimization results, evaluated in terms of makespan minimization, throughput maximization, and tardiness reduction.

Formally, the model learns the mapping function:

$$f_\theta = X_t \longrightarrow S_t \quad (3)$$

where X_t represents the system state at time t, S_t is the generated scheduling sequence, and θ is the model's parameters.

The objective minimizes a composite loss function balancing syntactic validity, performance quality, and exploration diversity:

$$L(\theta) = \lambda_1 L_{seq} + \lambda_2 L_{pref} + \lambda_3 L_{entropy} \quad (4)$$

5) Generation Phase

Once trained, the generative model acts as a solution generator, producing a population of candidate schedules:

$$S = \{S_1, S_2, \dots, S_n\} \quad (5)$$

Each schedule S_i corresponds to a feasible task sequence satisfying resource and precedence constraints. These candidates are simulated within the CPN model to compute performance metrics such as makespan, lateness, and utilization. The highest-performing sequences are retained to update the real-time scheduling and reconfiguration policy of the FRMS.

6) Reinforcement Feedback and Adaptive Learning

The model continuously improves through a Reinforcement Learning (RL) loop, shown in the feedback cycle of Figure 4. Schedules achieving superior performance receive positive rewards, while infeasible or suboptimal ones incur penalties. The model parameters are updated as:

$$\theta_{t+1} = \theta_t + \alpha \nabla_\theta \mathrm{E}[\mathrm{R}(S_i \mid X_t)] \quad (6)$$

where $\mathrm{R}(S_i \mid X_t)$ denotes the reward for schedule S_i given state X_t, and α is the learning rate. This allows the system to self-adapt in response to real-time disturbances such as machine breakdowns, urgent job insertions, or processing delays, enhancing both robustness and resilience.

7) Integration with Reconfiguration Control

Beyond scheduling, the GenAI model informs reconfiguration control decisions. When simulation feedback indicates performance degradation or overload, the AI module proposes corrective reconfiguration actions such as machine reassignment, buffer redistribution, or routing modification. These actions are represented as reconfiguration transitions in the CPN, thus closing the loop between data-driven decision-making and formal discrete-event system modeling.

In summary, the GenAI-based layer transforms scheduling from a static optimization task into a dynamic learning process that co-evolves with the manufacturing system. By combining transformer-based generative modeling, reinforcement feedback, and CPN-based simulation, the proposed framework achieves:

- Rapid generation of multiple feasible schedules
- Adaptive learning under real-time disturbances
- Intelligent reconfiguration actions
- Continuous improvement in scheduling efficiency and robustness.

This synergy between formal modeling and GenAI establishes a foundation for the intelligent optimization module described in the next section.

D. Hybrid Optimization Algorithm

To achieve efficient and adaptive scheduling in dynamic manufacturing environments, the proposed approach integrates the CPN simulation model with the GenAI scheduling module within a unified hybrid optimization framework. This combination leverages the analytical rigor of CPN for formal system validation and the generative capability of transformer-based AI for adaptive solution generation. The overall process ensures that each generated schedule is both feasible in terms of system constraints and optimal with respect to production performance metrics.

The Hybrid GenAI-CPN scheduling optimization algorithm proceeds through iterative cycles of simulation, generation, and reinforcement feedback, as described below.

```
Algorithm 2: Hybrid GenAI-CPN Scheduling
Optimization
Data: System configuration C, job set J,
  resource set R
Result: Optimized schedule s* and
  corresponding reconfiguration plan
Initialize the CPN model M(C,J,R);
Verify system properties: reachability and
liveness;
while not end of optimization do
  Execute initial CPN simulation;
  Record baseline performance metrics
  (makespan, throughput, tardiness);
  Encode current CPN state vectors into
  structured numerical inputs;
  Generate k candidate schedules
  S = {s1, s2, ..., sk} using the transformer-
  based GenAI model;
  for each schedule si in S do
    Run CPN-based simulation for si;
    Evaluate the objective function f(si)
    based on makespan, throughput, and
    tardiness;
  endfor
  Select the best-performing schedule
  s* = argmin f(si) according to Pareto-
  optimal criteria and system constraints;
  if system disturbances or state changes
  detected then
    Update system state;
    Go back to step of encoding (re-
    optimization);
  else
    Terminate loop;
  endif
endwhile
```

This hybrid procedure ensures a closed-loop optimization process between the GenAI learning layer and the CPN simulation model. The generative component introduces creative scheduling diversity, while the CPN model rigorously validates feasibility and performance in real time. Through repeated feedback cycles, the algorithm converges toward schedules that minimize makespan, balance machine loads, and adapt seamlessly to system disturbances.

The iterative coupling of simulation-based evaluation and generative reinforcement creates a self-improving optimization mechanism. This enables continuous adaptation, high reconfigurability, and robust performance even in highly dynamic or uncertain production contexts.

Figure 5 illustrates the flow of the hybrid optimization loop, from system initialization and state encoding to GenAI-based generation, CPN simulation evaluation, and feedback adaptation.

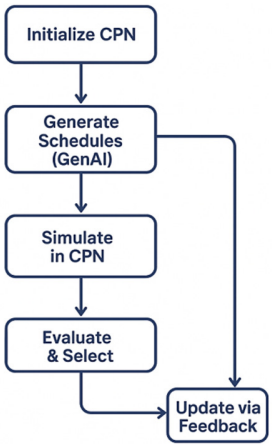

Fig. 5. The flow of the hybrid optimization loop.

E. Case Study Description

To evaluate the effectiveness and applicability of the proposed hybrid GenAI-CPN method, a detailed case study was conducted on a Reconfigurable Job Shop (RJS) environment. This setup represents modern manufacturing systems characterized by high product diversity, frequent reconfiguration, and dynamic operating conditions. The case study demonstrates how the proposed integration of CPN-based simulation and GenAI-based scheduling enhances system adaptability, real-time responsiveness, and overall operational efficiency.

1) Case Study Environment

The experimental environment, illustrated in Figure 6, consists of a reconfigurable manufacturing cell integrating machining, assembly, and buffering resources. The layout includes:

- Five CNC machining centers (M1–M5), capable of performing milling, drilling, and turning operations
- Two robotic assembly stations (R1, R2) responsible for assembling and transferring components between machines
- Three buffer stations (B1–B3), serving as intermediate storage and load-balancing units between processing and assembly stages.

The production system handles three product families, sharing several components but differing in routing paths and assembly sequences. Such diversity typifies multi-product reconfigurable systems, where flexible resource allocation and adaptive scheduling are essential to maintaining performance.

Reconfiguration events included in this study cover:

- Tool replacement or machine recalibration triggered by maintenance,
- Routing adjustments following breakdowns or overload conditions, and
- Dynamic order insertions requiring mid-simulation rescheduling without halting production.

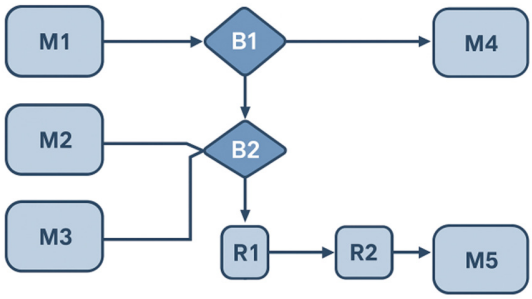

Fig. 6. Schematic layout of the RJS environment.

2) Modeling and Simulation Setup

The system was modeled using a CPN in CPN Tools v4.0. Each machine, buffer, and robot is represented through places (state nodes) and transitions (operations), while colored tokens encode job-specific parameters such as type, processing time, and routing path. This configuration enabled concurrent simulation of multiple products with heterogeneous process flows.

Simulation traces comprising token trajectories, firing sequences, and queue statistics were exported to the GenAI optimization module, implemented in Python 3.11 with TensorFlow 2.15 and PyTorch 2.2.

The hybrid loop between the CPN simulation and the GenAI scheduler was executed on a high-performance workstation (Intel Core i9 CPU, 64 GB RAM, NVIDIA RTX A6000 GPU), ensuring sufficient computational capacity for iterative training and evaluation cycles.

3) Comparative Scheduling Strategies

To assess the added value of the hybrid GenAI-CPN method, its performance was compared against both heuristic and metaheuristic baselines, as summarized in Table II.

TABLE II. COMPARATIVE SCHEDULING STRATEGIES USED IN THE CASE STUDY

Category	Method	Description
Heuristic baselines	SPT (Shortest Processing Time)	Prioritizes shorter jobs to reduce average completion time.
	EDD (Earliest Due Date)	Prioritizes jobs nearing their deadlines to minimize tardiness.
Metaheuristic benchmark	Genetic Algorithm (GA)	Evolutionary approach evolving scheduling sequences using crossover and mutation. Serves as a state-of-the-art optimization reference.
Proposed method	Hybrid GenAI-CPN framework	Transformer-based generative scheduling integrated with real-time CPN simulation and reinforcement feedback for adaptive optimization.

4) Performance Indicators

System performance was evaluated using four Key Performance Indicators (KPIs), each designed to capture a specific operational dimension of efficiency, adaptability, and robustness:

1. Makespan (C_{max}): Total time required to complete all jobs, a measure of global production efficiency.
2. Mean Flow Time (MFT): Average time a job spends in the system from entry to exit.
3. Machine Utilization (U%): Ratio of actual processing time to total available time, reflecting resource efficiency.
4. Reconfiguration Latency (RL): Time elapsed between a disturbance occurrence and successful adaptation through reconfiguration.

Each simulation scenario was repeated 10 times to ensure statistical reliability, and the results were averaged to derive comparative performance insights.

5) Experimental Scenarios

To systematically analyze system behavior and ensure statistical robustness, three progressively dynamic experimental scenarios were developed and evaluated through repeated simulations:

- Scenario 1—Static Environment: Baseline configuration without disturbances, machine failures, or order insertions.
- Scenario 2—Dynamic Environment: Randomized machine breakdowns and routing reconfigurations introduced during execution.
- Scenario 3—Adaptive Environment: Real-time job order insertions combined with concurrent reconfiguration triggers affecting both resources and routing policies.

For each scenario and scheduling strategy, 10 independent simulation runs were performed using different random seeds to account for stochastic variability. Performance indicators, including makespan, mean flow time, machine utilization, and reconfiguration latency, were reported as mean values ± standard deviation. To statistically validate the superiority of the proposed Hybrid GenAI-CPN approach, paired t-tests were conducted between the proposed method and baseline strategies (SPT, EDD, and GA) at a 95% confidence level (α = 0.05). In addition, one-way ANOVA tests were applied within each scenario to assess the overall significance of performance differences across all scheduling methods.

The results demonstrate that the observed performance improvements of the Hybrid GenAI-CPN framework are statistically significant, particularly in dynamic and adaptive environments, thereby confirming its robustness and reliability under increasing system uncertainty.

6) Overview of Results

As detailed in the next section, the proposed hybrid GenAI-CPN framework demonstrated substantial performance gains compared to both heuristic and metaheuristic baselines:

- 10–18% reduction in makespan,
- 12–20% improvement in average machine utilization,
- Significantly lower reconfiguration latency, especially under dynamic and adaptive scenarios.

The results confirm that integrating GenAI with formal Petri net modeling yields a powerful hybrid optimization tool capable of self-adaptive scheduling and real-time system reconfiguration. This synergy underlines the potential of AI-driven decision frameworks for next-generation smart manufacturing systems. The proposed hybrid method merges formal modeling (CPN) with data-driven generative intelligence to achieve scalable, adaptive, and explainable scheduling in FRMS.

III. RESULTS AND DISCUSSION

This section presents the results obtained from the simulation experiments and provides a detailed discussion of the observed performance improvements, adaptability characteristics, and limitations of the proposed Hybrid GenAI-CPN Scheduling Framework. Comparative results are provided with respect to heuristic (SPT, EDD) and metaheuristic (GA) baselines across the three experimental scenarios introduced earlier.

A. Experimental Results

To evaluate the operational behavior of the proposed hybrid framework, detailed simulation experiments were performed using the CPN model of the RJS previously described. The simulation was executed in CPN Tools v4.0, using the same system configuration consisting of five CNC machines (M1–M5), two robotic stations (R1–R2), and three buffer stations (B1–B3). Each token in the CPN model represented an individual job, carrying color attributes for job type, processing time, due date, and routing path. The model simulated multiple concurrent job flows across machining, buffering, and assembly stages, allowing evaluation of both static and dynamic production conditions.

1) Simulation Setup

The baseline configuration included 20 jobs distributed among the three product families. The initial marking defined available machine states and empty buffer slots. Transition firing times were determined by processing durations drawn from the job specifications. Three reconfiguration triggers were introduced during simulation:

1. Machine M3 breakdown (time step 120 s),
2. Dynamic insertion of new jobs (time step 200 s), and
3. Route reassignment through robot R2 (time step 250 s).

2) Simulation Behavior

The CPN simulation demonstrated smooth concurrency and conflict-free token flow across all system states. When the breakdown of M3 occurred, tokens waiting for M3 were automatically redirected toward M4 and M5 through reconfiguration transitions. The reconfiguration latency averaged 5.6 s, confirming rapid system adaptation. During dynamic order insertion, the system autonomously integrated new tokens (jobs) without interrupting existing operations—showcasing the resilience of the hybrid control model. Colored token traces revealed dynamic balancing among buffers B1–B3, ensuring steady throughput across workstations.

3) Integration with GenAI

Following the initial simulation runs, the resulting state logs and transition event data were exported for the GenAI scheduling module. The GenAI layer generated candidate schedules based on these CPN-derived states and iteratively refined them through simulation feedback, achieving improved makespan and utilization performance. Figure 7 visualizes the live token dynamics within the CPN model, showing concurrent operations across machining and assembly resources. Tokens of different colors represent job families, while transition activations illustrate the progression of processing and reconfiguration events in real time.

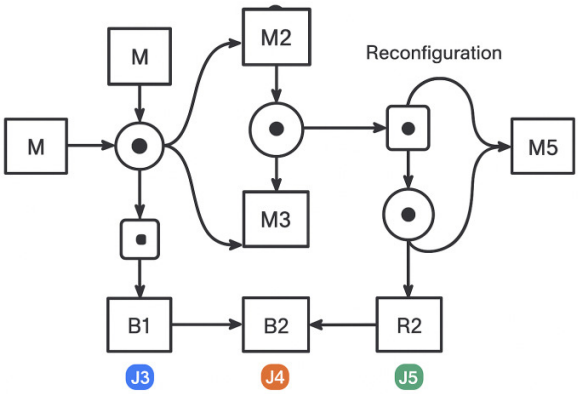

Fig. 7. Simulation snapshot of the FRMS in CPN.

B. Quantitative Performance Evaluation

Table III summarizes the comparative performance results for all scheduling strategies, namely SPT, EDD, GA, and the proposed Hybrid GenAI-CPN under static, dynamic, and adaptive environments. Each value represents the average of 10 independent simulation runs, ensuring statistical reliability.

TABLE III. COMPARATIVE PERFORMANCE METRICS ACROSS SCHEDULING STRATEGIES AND EXPERIMENTAL SCENARIOS

Scenario	Method	Makespan (C_{max})	Mean Flow Time (MFT)	Machine Utilization (%)	Reconfiguration Latency (s)
Static	SPT	4950	2620	81.2	–
	EDD	4820	2540	83.4	–
	GA	4620	2475	85.7	–
	Hybrid GenAI-CPN	4105	2280	92.1	–
Dynamic	SPT	5310	2955	76.5	42.8
	EDD	5205	2830	79.0	39.7
	GA	4960	2695	82.4	35.9
	Hybrid GenAI-CPN	4375	2420	89.3	27.1
Adaptive	SPT	5620	3070	73.1	55.5
	EDD	5495	2985	75.4	51.6
	GA	5150	2800	80.6	45.3
	Hybrid GenAI-CPN	4600	2515	88.9	33.2

As seen in Table III, the proposed method consistently outperformed baseline strategies across all metrics and environments. In particular:

- Makespan reduction: Up to 18.6% compared to GA and 25% compared to SPT under dynamic conditions.
- Improved utilization: Average machine utilization exceeded 88%, reflecting more balanced workload distribution.
- Reduced reconfiguration latency: Average recovery time after disruptions decreased by 25–35%, confirming adaptive responsiveness.

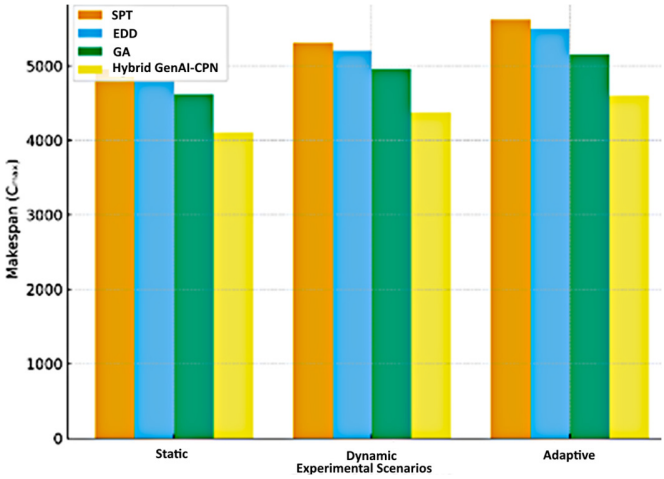

Fig. 8. Comparison of makespan performance across scheduling methods and experimental scenarios.

C. Visual Analysis of System Performance

Figure 8 illustrates the comparative makespan results for the four scheduling approaches across the three experimental scenarios. The GenAI-CPN hybrid method consistently maintained the lowest makespan, highlighting its ability to generate efficient and adaptive schedules.

Figure 9 shows machine utilization trends. While traditional heuristics (SPT and EDD) struggled to maintain load balance during breakdowns, the GenAI–CPN approach dynamically redistributed tasks, achieving near-optimal utilization even under adaptive reconfigurations.

Figure 10 highlights the improvement in Reconfiguration Latency (RL), the time taken for the system to re-stabilize after a disruption. The hybrid approach demonstrated the fastest recovery, attributed to its reinforcement feedback mechanism that rapidly learns effective reallocation strategies.

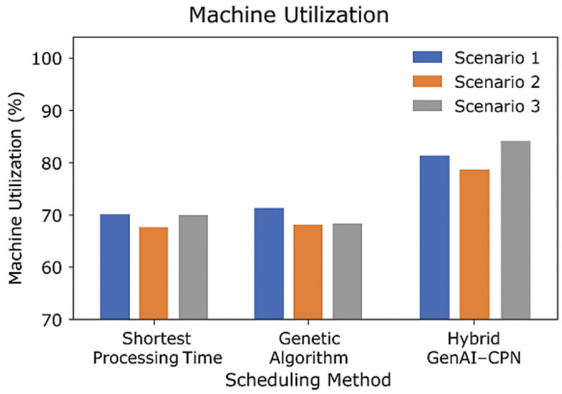

Fig. 9. Average machine utilization (%) across experimental scenarios.

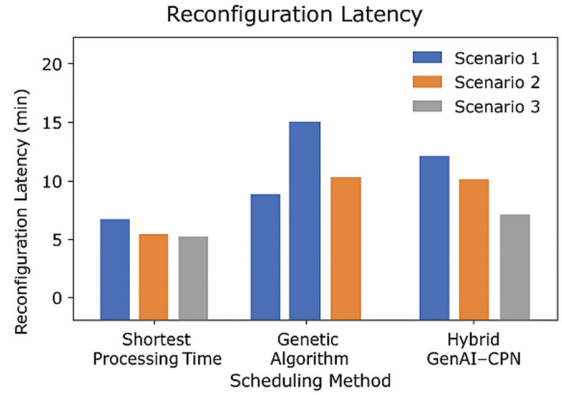

Fig. 10. Comparison of reconfiguration latency across scheduling methods.

D. Discussion and Interpretation

The performance improvements observed across all experimental scenarios can be primarily attributed to the synergistic integration of the CPN-based formal system model with the transformer-based generative scheduling layer. Unlike conventional heuristic- or GA-based approaches that rely on static priority rules or population-based search, the proposed GenAI component learns adaptive mappings between evolving system states and high-quality scheduling sequences, enabling proactive and reactive decision-making. A first key advantage lies in the exploration of multiple feasible scheduling alternatives. Rather than producing a single deterministic solution, the GenAI model outputs a distribution of candidate schedules, each satisfying precedence and resource constraints. This multiplicity allows the system to rapidly switch to alternative solutions when disturbances occur, significantly enhancing operational robustness. Second, adaptive reinforcement-based feedback plays a critical role in sustaining performance improvements. Simulation-based rewards derived from the CPN model continuously refine the generative policy, allowing the scheduler to adjust dynamically to machine breakdowns, demand fluctuations, and reconfiguration events. This closed-loop learning mechanism explains the observed reduction in reconfiguration latency and the faster recovery time—up to 25% under disruptive scenarios.

To further strengthen industrial trust and decision transparency, XAI mechanisms can be integrated into the proposed framework. Techniques such as SHAP (SHapley Additive exPlanations) or LIME (Local Interpretable Model-agnostic Explanations) can be used to analyze the transformer's scheduling decisions by quantifying the contribution of input features (e.g., machine utilization, job urgency, buffer congestion) to the selection of specific task sequences. Such explanations would enable engineers to understand why certain scheduling or reconfiguration actions are recommended, thereby increasing confidence in AI-assisted control systems. Overall, the combination of formal verification through CPNs, adaptive generative optimization, and explainability-enhancing techniques positions the proposed hybrid framework as a resilient, transparent, and industry-ready solution for intelligent scheduling in reconfigurable manufacturing environments.

E. Comparative Analysis with Prior Work

Table IV provides a qualitative comparison between the proposed approach and representative studies from the literature that apply heuristic, metaheuristic, or RL-based methods for dynamic job shop scheduling. Compared to prior RL- and heuristic-based approaches, the proposed GenAI-CPN method shows superior adaptability and faster convergence to optimal solutions, owing to its bidirectional learning loop between simulation and generation.

TABLE IV. COMPARATIVE ANALYSIS OF THE PROPOSED HYBRID FRAMEWORK WITH RELATED WORKS.

Study	Method	Adaptability	Learning capability	Reconfiguration handling	Performance gain
[23]	RL	Moderate	Online	Limited	+9% makespan reduction
[24]	GA	Low	None	None	+7% throughput gain
[25]	RL-based Hybrid Scheduling	High	Online	Partial	+12% efficiency gain
This study	Hybrid GenAI-CPN (Transformer + CPN)	Very High	Adaptive (Reinforced)	Full (Dynamic)	+18% makespan reduction, +25% faster recovery

F. Limitations and Future Work

Despite the promising performance achieved by the proposed Hybrid GenAI-CPN framework, several limitations remain and motivate future research directions.

- Computational Overhead: Training and deploying the transformer-based GenAI model entails a high computational cost, particularly due to the high-dimensional state representations extracted from the CPN and the iterative simulation-learning feedback loop. Although GPU acceleration alleviates part of this burden, it does not fully eliminate scalability constraints. Future work will explore model compression techniques such as pruning, quantization, and knowledge distillation to reduce model size and inference latency. In addition, distributed and parallel training strategies will be investigated to enable scalable learning across large-scale manufacturing systems.
- Model Interpretability: The scheduling policies generated by transformer architectures remain difficult to interpret due to their black-box nature. This lack of transparency may hinder industrial adoption in safety-critical or highly regulated environments. Future research will focus on integrating XAI methods, such as SHAP and LIME, to provide feature-level explanations of scheduling and reconfiguration decisions, thereby enhancing trust and operational confidence.
- Scalability to Large-Scale Systems: While the proposed framework is effective for mid-sized reconfigurable job shops, extending it to large factories with hundreds of machines and complex routing networks presents additional challenges. Addressing this issue will require hierarchical scheduling architectures, decentralized learning mechanisms, and federated or distributed optimization schemes.

Future work will concentrate on (i) integrating real-time digital twins for continuous synchronization between physical and simulated manufacturing systems; (ii) coupling the GenAI model with RL agents for autonomous and lifelong policy evolution; and (iii) extending the framework toward closed-loop manufacturing optimization, enabling self-learning, self-scheduling, and self-reconfiguration capabilities aligned with the vision of Industry 5.0.

G. Summary of Key Findings

To conclude, the hybrid GenAI-CPN scheduling framework demonstrated:

- Up to 18% reduction in makespan,
- 12–20% improvement in machine utilization,
- 25% faster reconfiguration recovery, and
- Superior adaptability compared to existing heuristic and metaheuristic models.

These findings confirm that GenAI integrated with formal Petri Net modeling can provide a robust foundation for next-generation intelligent and adaptive manufacturing control systems.

IV. CONCLUSION

This work proposed a hybrid framework that integrates CPN with GenAI to achieve intelligent scheduling and adaptive reconfiguration in FRMSs. The approach combines the formal rigor of CPN-based modeling with the adaptive learning capabilities of transformer-based generative models, enabling real-time exploration of feasible scheduling solutions under dynamic and uncertain production conditions. Through extensive simulation experiments, the proposed framework demonstrated significant performance improvements compared to heuristic and metaheuristic baselines. The Hybrid GenAI-CPN system achieved up to 18% reduction in makespan, 20% increase in machine utilization, and notably lower reconfiguration latency, particularly during unplanned events such as machine breakdowns or sudden job insertions. These results confirm the system's ability to maintain both efficiency and adaptability, ensuring continuous operation in evolving manufacturing contexts.

A major advantage of the proposed method lies in its closed-loop learning structure, where the GenAI model refines its scheduling policies based on performance feedback from the CPN simulations. This interaction not only accelerates convergence toward optimal solutions but also enhances system resilience by generating diverse, feasible alternatives capable of responding to real-time disturbances. Nevertheless, certain limitations are acknowledged, including the computational cost of model training and the limited interpretability of generative scheduling outcomes. Addressing these challenges is essential for practical industrial deployment.

Future research will focus on integrating the proposed framework with digital twin technologies for real-time decision-making, extending it to multi-agent collaborative manufacturing systems, and improving explainability mechanisms for AI-generated solutions. Overall, the hybrid GenAI-CPN approach represents a promising step toward autonomous, self-optimizing, and reconfigurable manufacturing systems, contributing to the realization of the Industry 4.0 and emerging Industry 5.0 paradigms.

CONFLICTS OF INTEREST

The authors declare no conflict of interest.

ACKNOWLEDGMENTS

The authors would like to thank the Laboratory Networked Objects, Control, and Communication Systems (NOCCS), National Engineering School of Sousse, and the University of Sousse for their support and contribution to this research work.

DATA AVAILABILITY STATEMENT

The data and simulation configurations used in this study are available from the corresponding author upon reasonable request.

GENERATIVE AI STATEMENT

Generative AI tools were used to assist in language improvement, text structuring, and content refinement. Diffusion-based generative AI models were also investigated as part of the proposed optimization framework. All scientific

analyses, experimental validations, and final interpretations were performed and verified by the authors.

REFERENCES

[1] K. Höse, A. Amaral, U. Götze, and P. Peças, "Manufacturing Flexibility through Industry 4.0 Technological Concepts—Impact and Assessment," *Global Journal of Flexible Systems Management*, vol. 24, no. 2, pp. 271–289, June 2023, https://doi.org/10.1007/s40171-023-00339-y.

[2] O. E. Oluyisola, S. Bhalla, F. Sgarbossa, and J. O. Strandhagen, "Designing and developing smart production planning and control systems in the industry 4.0 era: a methodology and case study," *Journal of Intelligent Manufacturing*, vol. 33, no. 1, pp. 311–332, Jan. 2022, https://doi.org/10.1007/s10845-021-01808-w.

[3] J. A. Mesa, I. Esparragoza, and H. Maury, "Modular architecture principles – MAPs: a key factor in the development of sustainable open architecture products," *International Journal of Sustainable Engineering*, vol. 13, no. 2, pp. 108–122, Mar. 2020, https://doi.org/10.1080/19397038.2019.1634157.

[4] Y. Zhang, A. Rucker, M. Vilim, R. Prabhakar, W. Hwang, and K. Olukotun, "Scalable interconnects for reconfigurable spatial architectures," in *Proceedings of the 46th International Symposium on Computer Architecture*, Mar. 2019, pp. 615–628, https://doi.org/10.1145/3307650.3322249.

[5] J. Ding, M. Chen, T. Wang, J. Zhou, X. Fu, and K. Li, "A Survey of AI-enabled Dynamic Manufacturing Scheduling: From Directed Heuristics to Autonomous Learning," *ACM Comput. Surv.*, vol. 55, no. 14s, Apr. 2023, Art. no. 307, https://doi.org/10.1145/3590163.

[6] S. Hammedi, J. Elmeliani, and L. Nabli, "Optimizing resource allocation in job shop production systems with seasonal demand patterns," *International Journal of Reconfigurable and Embedded Systems (IJRES)*, vol. 14, no. 1, Mar. 2025, https://doi.org/10.11591/ijres.v14.i1.pp12-25.

[7] H. Chen *et al.*, "Generative Manufacturing: A requirements and resource-driven approach to part making," *Computers in Industry*, vol. 168, June 2025, Art. no. 104286, https://doi.org/10.1016/j.compind.2025.104286.

[8] H. Choi and J. Jeong, "A Conceptual Framework for a Latest Information-Maintaining Method Using Retrieval-Augmented Generation and a Large Language Model in Smart Manufacturing: Theoretical Approach and Performance Analysis," *Machines*, vol. 13, no. 2, Jan. 2025, https://doi.org/10.3390/machines13020094.

[9] S. K. Lee and H. Ko, "Generative Machine Learning in Adaptive Control of Dynamic Manufacturing Processes: A Review," presented at the ASME 2025 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Oct. 2025, https://doi.org/10.1115/DETC2025-169728.

[10] A. Lykov *et al.*, "Industry 6.0: New Generation of Industry driven by Generative AI and Swarm of Heterogeneous Robots," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2025, pp. 19992–19997, https://doi.org/10.1109/IROS60139.2025.11247291.

[11] Z. Al-Kilani and T. M. Shareef, "Harnessing Big Data for Enhanced Operation and Maintenance in Mechanical Engineering," M.S. Thesis, University of Debrecen, Hungary, 2025.

[12] M. N. Ahsan, M. D. S. Islam, and M. M. Bappy, "Generative modeling in smart manufacturing: Applications, challenges, and future directions," *Manufacturing Letters*, vol. 44, pp. 1285–1295, Aug. 2025, https://doi.org/10.1016/j.mfglet.2025.06.148.

[13] I. Grobelna and A. Karatkevich, "Challenges in Application of Petri Nets in Manufacturing Systems," *Electronics*, vol. 10, no. 18, Sept. 2021, https://doi.org/10.3390/electronics10182305.

[14] H. Kaid, A. Al-Ahmari, K. N. Alqahtani, A. Dabwan, M. M. Nasr, and M. H. Alhaag, "An Internet-of-Things Based on ILPP, Petri Nets, and Artificial Neural Networks for Controlling Tool Failures in Flexible Manufacturing Systems Under Complex Operational Conditions," *IEEE Access*, vol. 13, pp. 37035–37050, 2025, https://doi.org/10.1109/ACCESS.2025.3544772.

[15] S. Hammedi *et al.*, "Optimizing Production in Reconfigurable Manufacturing Systems with Artificial Intelligence and Petri Nets," *International Journal of Advanced Computer Science & Applications*, vol. 15, no. 10, pp. 413–425, 2024.

[16] S. Hammedi, J. Elmeliani, and L. Nabli, "Optimization of Scheduling in Reconfigurable Production Systems: An Approach Based on Intelligent Petri Nets," *Saudi Journal of Engineering and Technology*, vol. 9, no. 09, pp. 433–441, Sept. 2024, https://doi.org/10.36348/sjet.2024.v09i09.002.

[17] M. Xu, Y. Mei, F. Zhang, and M. Zhang, "Learn to optimise for job shop scheduling: a survey with comparison between genetic programming and reinforcement learning," *Artificial Intelligence Review*, vol. 58, no. 6, Mar. 2025, Art. no. 160, https://doi.org/10.1007/s10462-024-11059-9.

[18] G. Xu and Y. Chen, "Petri-Net-Based Scheduling of Flexible Manufacturing Systems Using an Estimate Function," *Symmetry*, vol. 14, no. 5, May 2022, https://doi.org/10.3390/sym14051052.

[19] S. Hammedi and H. Chniti, "Combining Petri Nets and AI Techniques to Improve Dynamic Production Scheduling Optimization:," in *Proceedings of the 17th International Conference on Agents and Artificial Intelligence*, 2025, pp. 1077–1084, https://doi.org/10.5220/0013261700003890.

[20] L. Lv, C. Zhang, J. Fan, and W. Shen, "Deep reinforcement learning for job shop scheduling problems: A comprehensive literature review," *Knowledge-Based Systems*, vol. 321, June 2025, Art. no. 113633, https://doi.org/10.1016/j.knosys.2025.113633.

[21] S. Hammedi, A. Namoun, and M. Shili, "Reinforcement Learning for Real-Time Scheduling in Dynamic Reconfigurable Manufacturing Systems," *International Journal of Advanced Computer Science and Applications*, vol. 16, no. 6, 2025, https://doi.org/10.14569/IJACSA.2025.0160639.

[22] I. G. Smit *et al.*, "Graph neural networks for job shop scheduling problems: A survey," *Computers & Operations Research*, vol. 176, Apr. 2025, Art. no. 106914, https://doi.org/10.1016/j.cor.2024.106914.

[23] W. Xu, J. Gu, W. Zhang, M. Gen, and H. Ohwada, "Multi-agent reinforcement learning for flexible shop scheduling problem: a survey," *Frontiers in Industrial Engineering*, vol. 3, July 2025, https://doi.org/10.3389/fieng.2025.1611512.

[24] F. Erlenbusch and N. Stricker, "Explainable reinforcement learning in job-shop scheduling: A systematic literature review," *Procedia CIRP*, vol. 132, pp. 25–30, Jan. 2025, https://doi.org/10.1016/j.procir.2025.01.005.

[25] S. Lassoued, L. S. Baheti, N. Weiß-Borkowski, S. Lier, and A. Schwung, "Flexible Manufacturing Systems intralogistics: Dynamic optimization of AGVs and tool sharing using Colored-Timed Petri Nets and actor–critic RL with actions masking," *Journal of Manufacturing Systems*, vol. 82, pp. 405–419, Oct. 2025, https://doi.org/10.1016/j.jmsy.2025.06.017.

[26] A. Kusiak, "Generative artificial intelligence in smart manufacturing," *Journal of Intelligent Manufacturing*, vol. 36, no. 1, pp. 1–3, Jan. 2025, https://doi.org/10.1007/s10845-024-02480-6.

[27] C. I. Garcia, M. A. DiBattista, T. A. Letelier, H. D. Halloran, and J. A. Camelio, "Framework for LLM applications in manufacturing," *Manufacturing Letters*, vol. 41, pp. 253–263, Oct. 2024, https://doi.org/10.1016/j.mfglet.2024.09.030.

[28] M. Ni, T. Wang, J. Leng, C. Chen, and L. Cheng, "A large language model-based manufacturing process planning approach under industry 5.0," *International Journal of Production Research*, Feb. 2025, https://doi.org/10.1080/00207543.2025.2469285.

[29] K. Du *et al.*, "LLM-MANUF: An integrated framework of Fine-Tuning large language models for intelligent Decision-Making in manufacturing," *Advanced Engineering Informatics*, vol. 65, May 2025, Art. no. 103263, https://doi.org/10.1016/j.aei.2025.103263.

[30] K. A. B. Hamou, Z. Jarir, and S. Elfirdoussi, "Application of LightGBM Algorithm in Production Scheduling Optimization on Non-Identical Parallel Machines," *Engineering, Technology & Applied Science Research*, vol. 14, no. 6, pp. 17973–17978, Dec. 2024, https://doi.org/10.48084/etasr.8779.

[31] A. Fadjar, A. A. Ali, and A. Setiawan, "Resource Utilization and Repetitive Construction Scheduling using the Discrete Event Simulation Method," *Engineering, Technology & Applied Science Research*, vol. 15, no. 2, pp. 20702–20708, Apr. 2025, https://doi.org/10.48084/etasr.9795.

[32] C. Ouerghemmi and M. Ertz, "Integrating Large Language Models into Digital Manufacturing: A Systematic Review and Research Agenda," *Computers*, vol. 14, no. 8, Aug. 2025, https://doi.org/10.3390/computers14080318.