Hybrid Optimization-Rule Framework for Real-Time Job Shop Co-Scheduling

Doyoung Kim, Woojin Jun, KyeongMin Yeom, and Moon Gi Seok
Dept. of Computer Science and AI Engineering, Dongguk University, Seoul, Republic of Korea
Email: {doyoung.kim, woojin.jun, kyeongmin.yeom, mgseok}@dgu.ac.kr

Abstract— **The Flexible Job Shop Co-Scheduling Problem (FJCSP), which integrates machine operations with Automated Guided Vehicle (AGV) routing, represents a highly complex online scheduling challenge under stringent real-time constraints. Traditional optimization methods, while capable of producing high-quality schedules, are computationally prohibitive for online decision-making, whereas rule-based methods offer rapid responsiveness but yield unstable performance. To address this trade-off, we propose a reinforcement learning (RL)-driven hybrid framework that selectively applies NSGA-II optimization to a limited set of critical operations, while dispatching rules govern the remaining tasks. An RL agent dynamically adjusts the number of optimized operations to balance solution quality against computational efficiency. Experimental validation in a digital twin environment demonstrates that the proposed approach significantly reduces due date violations and makespan compared to pure rule-based scheduling, while achieving performance close to full NSGA-II optimization with more than 50**

I. Introduction

The integration of Flexible Manufacturing Systems (FMS) with Automated Guided Vehicles (AGVs) transforms the classical Flexible Job Shop Scheduling Problem (FJSP) into the more complex *Flexible Job Shop Co-Scheduling Problem (FJCSP)*, where machine operations and AGV routing must be coordinated simultaneously. This NP-hard problem is critical in modern production environments such as semiconductor manufacturing and automated container terminals.

Although extensive studies have addressed FJSP and its variants, most existing approaches suffer from a fundamental limitation: they operate offline and cannot adapt to real-time changes. As a result, they are unsuitable for highly dynamic shop-floor environments where decisions must be made within milliseconds. Early methods using Petri nets [1] or MILP [2] had limited scalability. Metaheuristics [3], [4] improved performance but remain offline. However, even the most advanced algorithms still require prohibitive computation time, which is impractical for real-time applications.

We propose a fundamentally different approach: instead of optimizing all operations, we adaptively select at most $N_{\max}$ critical operations for multi-objective optimization, while applying efficient dispatching rules to the rest. The key research question is: *Can online co-scheduling be achieved by dynamically adjusting $N_{\max}$ to balance solution quality with real-time responsiveness?*

Our *hybrid optimization-rule framework* employs reinforcement learning to determine $N_{\max}$ based on the observed system state. Critical operations are optimized with NSGA-II (hard commitments), while routine operations follow dispatching rules (soft assignments). This approach preserves solution quality close to full NSGA-II optimization while reducing computation time by more than half, demonstrating its potential to satisfy real-time scheduling requirements.

The main contributions of this paper are:

- A **hybrid optimization-rule framework** that selectively applies NSGA-II to critical operations while using rules for others
- A **reinforcement learning agent** that adaptively learns $N_{\max}$ selection based on system state
- A **dual commitment strategy** ensuring reliability for critical operations and flexibility for routine ones
- Experimental validation in a digital twin simulation, showing significant quality improvement over rules and more than 50% runtime reduction compared to NSGA-II

The remainder of this paper is organized as follows. Section III defines the FJCSP formulation. Section II reviews related work. Section IV presents the proposed framework. Section V reports experimental results, followed by discussion and conclusion in Section **??**.

II. Related Work

The Flexible Job Shop Scheduling Problem (FJSP), when integrated with Automated Guided Vehicles (AGVs), becomes the Flexible Job Shop Co-Scheduling Problem (FJCSP). This variant is more complex than the classical FJSP, and prior studies can be grouped into model-based, metaheuristic, hybrid, AGV-specific, and learning-based approaches.

A. Model-based approaches

Early works modeled machine–AGV scheduling with Petri nets [1] and MILP formulations [2]. While exact, these methods suffer from severe scalability issues, limiting their use in dynamic shop floors.

B. Metaheuristic approaches

To improve scalability, metaheuristics such as genetic algorithms [3], tabu search [4], and more recent designs like CSSA [5] and EDA–ACO hybrids [6] have been proposed. These methods produce high-quality solutions but require heavy iteration, making them unsuitable for real-time scheduling.

C. Hybrid algorithms

Hybrid approaches aim to balance efficiency and quality. For example, Lacomme et al. [7] integrated machine and AGV scheduling in a unified framework, while Meng et al. [8] combined CP models with metaheuristics. However, their runtimes—seconds to tens of seconds—remain prohibitive for online use.

D. AGV-focused surveys

Surveys such as Qiu et al. [9] and Vis [10] reviewed AGV routing and control strategies, emphasizing the need to integrate transport resources with production scheduling.

E. Learning-based methods

Recently, reinforcement learning has been explored to adapt under dynamic conditions. Examples include deep RL frameworks [11], DQN-based cooperative scheduling [12], and MARL with efficient action decoding [13]. Although more responsive, these methods still struggle with overly large action spaces.

F. Limitations and motivation

In summary, metaheuristics and hybrids yield quality solutions but are too slow, while RL improves adaptability but faces complexity issues. To bridge this gap, we propose a hybrid optimization–rule framework: optimize at most $N_{\max}$ critical operations using NSGA-II, while applying dispatching rules (e.g., ECT, FCFS) to the rest, ensuring both responsiveness and solution quality in real-time settings.

III. Problem Formulation

We consider the Flexible Job Shop Co-Scheduling Problem (FJCSP) involving a set of jobs $\mathcal{J} = \{J_1, \dots, J_{N_J}\}$, a set of machines $\mathcal{M} = \{M_1, \dots, M_{N_M}\}$, and a set of AGVs $\mathcal{A} = \{A_1, \dots, A_{N_A}\}$. Each job J_j consists of a sequence of operations $O_{j,1}, \dots, O_{j,N_j^{\text{opr}}}$ that must be processed in order. Each operation requires processing on one of the eligible machines and may require transportation by an AGV to the next machine. Machines can handle at most one operation at a time, and AGVs can transport at most one job at a time, with transportation consisting of *no-load* and *loaded* stages. Transportation times are assumed to be deterministic and depend only on machine locations. The decision variables are as follows: $X_{j,i,m} \in \{0,1\}$ indicates whether operation $O_{j,i}$ is processed on machine M_m; $Z_{j,i,k} \in \{0,1\}$ indicates whether operation $O_{j,i}$ is transported by AGV A_k; $S^p_{j,i}, C^p_{j,i}$ denote the start and completion times of processing; and $S^t_{j,i}, C^t_{j,i}$ denote the start and completion times of transportation.

The scheduling constraints are defined as follows. Each operation must be assigned to exactly one machine and one AGV:

$$\sum_{m \in \mathcal{M}} X_{j,i,m} = 1, \quad \forall j, i, \tag{1}$$

$$\sum_{k \in \mathcal{A}} Z_{j,i,k} = 1, \quad \forall j, i. \tag{2}$$

Job precedence must be respected, such that each operation can only start after the transport of its predecessor is completed:

$$S^p_{j,i} \geq C^t_{j,i-1}, \quad \forall j, i > 1. \tag{3}$$

Capacity constraints ensure that machines and AGVs cannot execute more than one task simultaneously:

$$S^p_{j,i} \geq C^p_{j',i'} \text{ if } X_{j,i,m} = X_{j',i',m} = 1, \tag{4}$$
$$S^t_{j,i} \geq C^t_{j',i'} \text{ if } Z_{j,i,k} = Z_{j',i',k} = 1. \tag{5}$$

The optimization objective is formulated as a multi-objective problem:

$$F_1 = C_{\max} = \max_j C^p_{j,N_j^{\text{opr}}}, \qquad \text{Makespan} \tag{6}$$

$$F_2 = \sum_{j,i} \text{dist}_{j,i}, \qquad \text{Total AGV travel distance} \tag{7}$$

$$F_3 = \sum_{j,i} \max\{0, C^p_{j,i} - d_{j,i}\}, \quad \text{Due date violation} \tag{8}$$

$$\min\{F_1, F_2, F_3\}. \tag{9}$$

Here, $d_{j,i}$ is the due date derived from release time and processing requirements, and $\text{dist}_{j,i}$ is the AGV travel distance. F_1 penalizes violations of job due dates, F_2 minimizes the overall completion time, and F_3 minimizes transportation cost. Note that the capacity constraints are conceptual; feasible schedules are generated by an SSGS-style decoder with disjunctive no-overlap in simulation.

We adopt the following assumptions to keep the model tractable. Each machine and AGV can process or transport only one task at a time. Travel times between machines are deterministic and known. Machines and AGVs may be initially occupied, with given availability times. Jobs are finite and known within the scheduling horizon.

Critically, we assume: (1) access to a *discrete-event simulator* that can handle mixed constraint specifications—it enforces specific machine-AGV assignments for selected operations (fixed mappings) while applying a pre-defined default rule for others; (2) the *dispatching rules* are defined as follows: for job–machine matching, the *Earliest Completion Time (ECT) rule* is used, which selects the machine that can complete the operation the earliest by considering both the input queue and the remaining processing time of the current task; for AGV–job matching, the *First-Come, First-Served (FCFS) rule* is applied. This simulator capability and dispatching rule policy are essential for the hybrid execution approach. The RL agent observes the current state and determines the appropriate $N_{\max}$ value, while NSGA-II optimizes assignments for the selected critical operations through simulation-based evaluation with the dispatching rules handling the rest.

Traditional offline methods optimize over the entire scheduling horizon to approximate a global optimum, which incurs prohibitive computational cost. In contrast, our formulation restricts optimization to at most $N_{\max}$ candidate operations, determined by the RL agent. This strategy reduces the search space, enabling fast online scheduling while ensuring that deviations from the global optimum remain controlled. Thus, the formulation bridges offline global optimization and online adaptive scheduling.

IV. Proposed Method

We adopt an *event-driven online co-scheduling policy* that coordinates machine and AGV operations under real-time conditions. At each decision epoch t, the controller observes only the realized shop-floor state (running operations, machine/AGV availability, released jobs) and selects at most $N_{\max}$ *candidate operations* for integrated scheduling. This *non-anticipative, incremental, and deadline-bounded* procedure meets the standard definition of *online scheduling*.

A. Hybrid Optimization-Rule Co-Scheduling with $N_{\max}$-Bounded Operations

The core innovation lies in a *hybrid optimization-rule approach* that combines the solution quality of multi-objective optimization with the computational efficiency of dispatching rules. At each decision epoch t, the framework operates as follows:

1. Critical Operation Selection: Instead of optimizing all pending operations, the system identifies at most $N_{\max}$ critical operations based on a scored ranking with diversity:

$$G(o) = \alpha \cdot process(o) + \beta \cdot move(o) + \gamma \cdot urgency(o) + \delta \cdot competition(o) \tag{10}$$

a) Average Processing Time:

$$process(o) = \bar{p}(o) = \frac{1}{|M_{i,j}|} \sum_{m \in M_{i,j}} p_{i,j,m} \tag{11}$$

Here, $\bar{p}(o)$ denotes the average processing time of operation $o = (i,j)$ over the candidate machine set $M_{i,j}$, and $p_{i,j,m}$ is the processing time of o on machine $m \in M_{i,j}$.

b) Average Transportation Time:

$$move(o) = \bar{\tau}(o) = \begin{cases} \frac{1}{|M_{i,j}|} \sum_{m \in M_{i,j}} \tau_{loc(i,j) \to m^*}, \\ \text{for currently available operations} \\ \frac{1}{|\Pi_{j,i}|} \sum_{\pi \in \Pi_{j,i}} \tau_{\text{src}(\pi) \to m_i(\pi)}, \\ \text{for future operations} \end{cases} \tag{12}$$

Here, $\bar{\tau}(o)$ represents the average transportation time of operation $o = (i,j)$. If the current location $loc(i,j)$ is known, it is defined as the average transportation time to the candidate machines. For future operations with uncertain previous/next positions, it is computed as the expected transportation time averaged over the feasible route set $\Pi_{j,i}$.

c) Urgency (Due-Date Based):

$$RPT_{i,j}(t) = \sum_{h=j+1}^{N_j^{opr}} \left(\bar{p}_{i,h} + \bar{\tau}_{i,h}\right) \tag{13}$$

$$LST_{i,j}(t) = due_i - RPT_{i,j}(t) - \bar{p}_{i,j} \tag{14}$$

$$urgency(o,t) = \max(0,\ t - LST_{i,j}(t)) \tag{15}$$

Here, $RPT_{i,j}(t)$ is the remaining processing time after operation $o = (i,j)$, and $LST_{i,j}(t)$ is the latest start time to meet the due date due_i. The urgency measure $urgency(o,t)$ is defined as the deviation of the current time t from $LST_{i,j}(t)$, indicating how critical it is to start the operation to avoid due-date violations.

d) Competition:

$$comp_m = |\{o' \in C : m \in M_{o'}\}| \tag{16}$$

$$competition(o) = \frac{1}{|M_{i,j}|} \sum_{m \in M_{i,j}} comp_m - 1 \tag{17}$$

Here, $comp_m$ represents the number of competing operations in the candidate set C that also require machine m. The competition score $competition(o)$ is the average competition intensity across the candidate machine set $M_{i,j}$, reflecting the expected contention level of operation o.

e) : Finally, the selection process employs ε-greedy replacement and type-wise quotas to ensure that a diverse set of operations is selected as critical candidates.

2. Hybrid Execution Strategy:

- **NSGA-II Optimization** (for $N_{\max}$ operations): Multi-objective optimization determines optimal machine-AGV assignments, considering all constraints and objectives. These assignments are *hard commitments* that must be strictly enforced.
- **Rule-based Dispatching** (for remaining operations): All other operations follow deterministic rules (SPT, EDD, or FCFS) with minimal computational overhead. These assignments are *soft* and can be revised in subsequent epochs.

This hybrid approach ensures that critical decisions receive thorough optimization while maintaining tractability. The complexity reduces from $O(N_J \times N_M \times N_A)$ to $O(N_{\max} \times N_M \times N_A) + O(N_{\text{remaining}})$, where the second term has linear complexity due to rule-based assignment.

B. RL-based $N_{\max}$ Policy Learning

The RL agent learns to dynamically adjust $N_{\max}$, effectively controlling the balance between optimization and rule-based dispatching. The augmented state representation includes:

$$s_t = \{\, \text{machine features},\ \text{AGV features},\ \text{global progress} \,\}. \tag{18}$$

- **Machine Features**: For each machine, we consider (i) the total processing time of the input queue, (ii) the remaining processing time of the job currently in service, and (iii) the number of jobs waiting in the output queue. Their minimum, mean, and maximum values are extracted and normalized to provide aggregate descriptors of machine load.
- **AGV Features**: The numbers of AGVs in delivery, fetch, and idle states are included, along with the average fleet utilization rate.

- **Global Progress**: The overall job completion ratio and the average machine utilization are incorporated to summarize system-level progress.

This yields a 15-dimensional feature vector that compactly represents operation queues, resource utilization, and transportation status as the input to the RL agent.

The action space consists of discrete choices: $N_{\max} \in \{4, 8, 12, 16, 24\}$, where:

- Small $N_{\max}$ (4-8): Mostly rule-based execution, suitable for stable conditions
- Medium $N_{\max}$ (12-16): Balanced hybrid mode for normal operations
- Large $N_{\max}$ (24): Optimization-heavy mode for critical situations

The reward function is designed to learn the optimal trade-off:

At each decision epoch t, the reward is defined as a penalty over four terms: (i) due-date violation, (ii) degradation of the makespan upper bound relative to a baseline rule, (iii) AGV travel distance, and (iv) the computation time required to run NSGA-II on the selected $N_{\max}$ operations.

$$r_t = -\Big(\lambda_{\text{viol}} \cdot \text{Viol}_t + \lambda_C \cdot \text{Gap}_t + \lambda_{\text{move}} \cdot \text{Move}_t + \lambda_{\text{time}} \cdot t^{\text{NSGA}}\Big), \tag{19}$$

where $\lambda_{\text{viol}}, \lambda_C, \lambda_{\text{move}}, \lambda_{\text{time}} \geq 0$ are tunable weights and the terms are defined below:

- Viol_t: the total due-date violation at epoch t, e.g., $\sum_{j,i} \max\{0, C^{p}_{j,i} - d_{j,i}\}$.
- Gap_t: the excess makespan upper bound compared to a baseline rule solution π^0:

  $$\text{Gap}_t = \max\{ C^{UB}_{\max}(t \mid a_t) - C^{UB}_{\max}(t \mid \pi^0), 0\},$$

  where $C^{UB}_{\max}$ denotes the simulator-estimated makespan upper bound under partial commitments.
- Move_t: AGV travel distance at epoch t, measured by total travel distance or travel time.
- t^{NSGA}: the elapsed computation time to execute NSGA-II on the selected set of $N_{\max}$ critical operations at epoch t.

The key insight is that Viol_t heavily penalizes poor decisions on critical operations, teaching the agent to increase $N_{\max}$ when quality matters. Conversely, the normalized computation time term t^{NSGA} encourages smaller $N_{\max}$ when rules suffice. Through training, the agent learns to recognize patterns indicating when optimization effort is worthwhile versus when rules are adequate.

C. Incremental Execution with Hybrid Commitment

Following the online scheduling paradigm, we employ a *simulation-based optimization* approach where $N_{\max}$ critical operations receive NSGA-II optimization. The system maintains a *dispatching rule* that is consistently applied to all non-optimized operations throughout both fitness evaluation and actual execution.

The key aspects of our simulation-based hybrid approach:

1. Default Rule Policy:

- The system maintains a pre-defined *default dispatching rule*.
- This default rule is consistently applied throughout:
  - During NSGA-II fitness evaluation (for non-optimized operations)
  - During actual execution (for operations not in $\mathcal{C}$)
- The same default rule ensures consistency between simulation and reality

2. Hybrid Execution Policy in Simulation:

- When the simulator processes an operation o:

$$\text{Assignment}(o) = \begin{cases} \pi^{\star}[o] & \text{if } o \in \mathcal{C} \text{ (fixed by NSGA-II)} \\ \mathcal{D}(o) & \text{otherwise (default rule)} \end{cases} \tag{20}$$

- This hybrid policy ensures that optimized decisions are strictly enforced while maintaining efficiency for routine operations

3. Critical Assumptions:

- **Simulator capability**: The discrete-event simulator can distinguish between fixed mappings and default-rule operations
- **Rule consistency**: The same default rule $\mathcal{D}$ is used in both fitness evaluation and actual execution
- **Deterministic evaluation**: Given the same fixed mappings and default rule, the simulator produces consistent results

4. Mapping Types:

- **Fixed Mappings**: NSGA-II optimized machine-AGV assignments for $N_{\max}$ operations are immutable and override the default rule
- **Default Rule Application**: All other operations follow the pre-defined default rule $\mathcal{D}$

This default rule policy ensures predictable behavior for non-critical operations while allowing optimization to focus on high-impact decisions.

V. Experiments

In this section, the performance of the proposed RL-based hybrid online co-scheduling method is validated in a digital twin simulation environment. Comparisons are made against two baselines (rule-based scheduling and NSGA-II) and the proposed RL–NSGA-II approach. Performance is evaluated over 10 independent simulation trials, reporting the minimum, mean, and maximum values of makespan, AGV travel distance, due date violations, and optimization runtime.

A. Experimental Setup and Parameters

Experiments are carried out using a scenario generator that constructs a shop floor with multiple jobs, machines, and AGVs. Each job follows predefined precedence constraints, and each AGV is restricted to transporting at most one job at a time. Rule-based scheduling employs the ECT policy for machine–job matching and the FCFS policy for AGV–job matching.

TABLE I: Experimental scenario and NSGA-II parameters

Parameter	**Value**
Scenario parameters	
Number of jobs	40
Operations per job	3–5
Number of machines	8
Number of AGVs	5
Processing times	Uniform[20, 40]
Transfer times	Uniform[5, 10]
Release times	10-job batches in [0, 590]
Due times	Release time + processing + [0, 200]
NSGA-II parameters	
Population size	100
Generations	200
Crossover probability	0.7
Mutation probability	0.2

B. Comparison Methods

We compare three scheduling strategies:

1) **Rules**: rule-based scheduling only (ECT for machine–job matching; FCFS for AGV–job matching)
2) **NSGA-II**: NSGA-II applied to all operations
3) **RL + NSGA-II (Proposed)**: RL selects the candidate set for NSGA-II optimization

C. Evaluation Metrics

Each scenario is executed 10 times under identical conditions. We analyze:

- **Makespan(F1)**: overall job completion time
- **AGV Travel Distance(F2)**: total travel distance of AGVs
- **Due Date Violation(F3)**: total tardiness beyond due dates
- **Optimization Runtime(F4)**: computation time consumed by NSGA-II generations

For each metric, we report the minimum, mean, and maximum values to assess the balance between solution quality and computational efficiency.

D. Results and Analysis

Table II and Figure 1 summarize the performance of the three scheduling strategies—rule-based scheduling, NSGA-II optimization, and the proposed RL+NSGA-II hybrid. Results are reported as the minimum, mean, and maximum values over ten independent simulation trials.

TABLE II: Performance comparison of scheduling methods (min/mean/max over 10 runs).

Method	**F1**	**F2**	**F3**	**F4**
	879.50	1902.00	3147.10	0.00
Rules	1083.42	1994.00	10851.72	0.00
	1360.05	2184.00	20536.10	0.00
	685.10	1873.00	1901.90	71.31
NSGA-II	801.82	1956.60	5289.42	94.42
	984.00	2019.00	10424.30	112.72
	756.10	1899.00	2561.40	35.90
RL+NSGA-II	851.98	1969.40	5781.45	41.88
	987.00	2187.00	11544.60	53.97

a) Makespan (F1).: The rule-based approach yields the largest average makespan (1083.4), indicating inefficiency in workload balancing. NSGA-II achieves the shortest average makespan (801.8), serving as the quality upper bound. RL+NSGA-II achieves 851.9 on average, shorter than Rule and close to NSGA-II.

b) AGV Travel Distance (F2).: All three methods produce similar results in the range of 1900–2000 on average, and no significant differences are observed. The minimum–maximum ranges are also comparable across the three methods.

c) Due Date Violations (F3).: The rule-based method shows a large spread between minimum and maximum values, reaching more than 20,000 violations in some scenarios, which indicates instability. NSGA-II produces a narrower range and more consistent performance. RL+NSGA-II achieves a comparable average to NSGA-II while significantly reducing the violation scale compared to Rule.

d) Optimization Runtime (F4).: The rule-based method does not apply to this metric and is excluded from the results. NSGA-II requires an average of 94 s (up to 113 s), leading to substantial computational overhead. RL+NSGA-II reduces this to 41.9 s on average, less than half of NSGA-II, making it more suitable for satisfying online decision-making constraints.

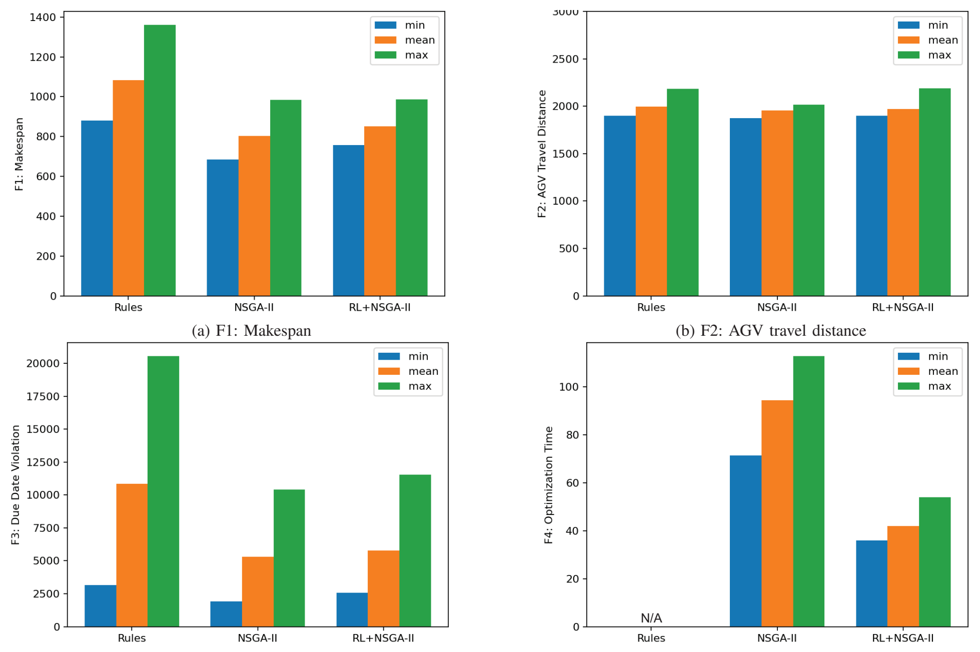
VI. Discussion and Conclusion

The experiments demonstrate that the proposed RL+NSGA-II hybrid framework achieves a balanced trade-off between rule-based scheduling and full optimization.

The **rule-based approach** ensures rapid computation but exhibits high performance variance, particularly in makespan and due date violations. This instability shows that heuristics alone cannot reliably handle diverse shop-floor conditions. Conversely, **NSGA-II** consistently delivers high-quality schedules with minimal due date violations, serving as a performance upper bound, yet its high computation time renders it impractical for real-time decision-making.

By contrast, the **RL+NSGA-II hybrid** attains solution quality close to NSGA-II while reducing computation time by more than half. Through adaptive selection of a limited set of critical operations, the RL agent minimizes unnecessary optimization and maintains responsiveness within decision deadlines. This result confirms the framework's effectiveness in resolving the inherent *quality–efficiency trade-off* of online co-scheduling.

Overall, the framework complements the limitations of single-strategy methods by combining responsiveness with stable solution quality, making it promising for deployment in real manufacturing environments. Some quality loss may occur under extreme conditions, which highlights the need for further validation in larger-scale or highly uncertain environments. Future work will extend the approach to more complex scenarios, explore hierarchical rule integration, and conduct industrial testbed experiments.



(a) F1: Makespan

(b) F2: AGV travel distance

(c) F3: Due date violation

(d) F4: Optimization runtime

Fig. 1: Performance comparison of Rules, NSGA-II, and RL+NSGA-II across four metrics: (a) makespan, (b) AGV travel distance, (c) due date violation, and (d) optimization runtime.

ACKNOWLEDGMENT

This research was supported by the "Regional Innovation System & Education (RISE)" program through the Seoul RISE Center, funded by the Ministry of Education (MOE) and the Seoul Metropolitan Government (2025-RISE-01-007-05), and by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Ministry of Science and ICT (MSIT), Republic of Korea, under the Artificial Intelligence Convergence Innovation Human Resources Development program (IITP-2025-RS-2023-00254592) and the IITP-ICAN (ICT Challenge and Advanced Network of HRD) program (IITP-2025-RS-2023-00260248).

REFERENCES

[1] Lee, D. Y., & DiCesare, F. (1994). Integrated scheduling of flexible manufacturing systems employing automated guided vehicles. *IEEE Transactions on Industrial Electronics*, 41(6), 602–610. https://doi.org/10.1109/41.334585

[2] Bilge, U., & Ulusoy, G. (1995). A time window approach to simultaneous scheduling of machines and material handling system in an FMS. *Operations Research*, 43(6), 1058–1070. https://doi.org/10.1287/opre.43.6.1058

[3] Ulusoy, G., Sivrikaya-Serifoglu, F., & Bilge, U. (1997). A genetic algorithm approach to the simultaneous scheduling of machines and automated guided vehicles. *Computers & Operations Research*, 24(4), 335–351. https://doi.org/10.1016/S0305-0548(96)00065-6

[4] Zheng, B., Xiao, T., & Seo, Y. (2014). A tabu search algorithm for simultaneous machine/AGV scheduling problem. *International Journal of Production Research*, 52(10), 3074–3090. https://doi.org/10.1080/00207543.2013.857791

[5] Sun, C., Li, D., Wang, Y., Tan, Y., Zhang, F., & Zhang, J. (2023). Chaotic sparrow search algorithm based scheduling for flexible job shop with automatic guided vehicle. In *Proc. International Conference on Information Science, Parallel and Distributed Systems (ISPDS)* (pp. 580–586). IEEE. https://doi.org/10.1109/ISPDS58840.2023.10235475

[6] Xin, B., Lu, S., Wang, Q., Deng, F., Shi, X., Cheng, J., & Kang, Y. (2024). Simultaneous scheduling of processing machines and automated guided vehicles via a multi-view modeling-based hybrid algorithm. *IEEE Transactions on Automation Science and Engineering*, 21(3), 4753–4768. https://doi.org/10.1109/TASE.2023.3301656

[7] Lacomme, P., Larabi, M., & Tchernev, N. (2013). Job-shop based framework for simultaneous scheduling of machines and automated guided vehicles. *International Journal of Production Economics*, 143(1), 24–34. https://doi.org/10.1016/j.ijpe.2010.07.012

[8] Meng, L., Cheng, W., Zhang, C., Gao, K., Zhang, B., & Ren, Y. (2025). Novel CP models and CP-assisted meta-heuristic algorithm for flexible job shop scheduling benchmark problem with multi-AGV. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*. https://doi.org/10.1109/TSMC.2025.3604355

[9] Qiu, L., Hsu, W. J., Huang, S. Y., & Wang, H. (2002). Scheduling and routing algorithms for AGVs: A survey. *International Journal of Production Research*, 40(3), 745–760. https://doi.org/10.1080/00207540110091712

[10] Vis, I. F. A. (2006). Survey of research in the design and control of automated guided vehicle systems. *European Journal of Operational Research*, 170(3), 677–709. https://doi.org/10.1016/j.ejor.2004.09.020

[11] Lei, Z., Cao, Z., Zhang, J., Song, W., & Zhang, J. (2022). A multi-action deep reinforcement learning framework for flexible job-shop scheduling problem. *IEEE Transactions on Industrial Informatics*, 18(6), 3882–3892. https://doi.org/10.1109/TII.2021.3117937

[12] Dong, X., Wan, G., & Zeng, P. (2024). Flexible job shop machines and AGVs cooperative scheduling on the basis of DQN algorithm. In *Proc. IEEE Int. Conf. Advanced Information Management, Communications, Electronic and Automation Control (IMCEC)* (pp. 1808–1815). IEEE. https://doi.org/10.1109/IMCEC59810.2024.10575192

[13] Li, Y., Wang, Q., Li, X., Gao, L., Fu, L., Yu, Y., & Zhou, W. (2025). Real-time scheduling for flexible job shop with AGVs using multiagent reinforcement learning and efficient action decoding. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 55(3), 2120–2135. https://doi.org/10.1109/TSMC.2024.3520381