Adaptive SKU Allocation in Mattress Manufacturing via Multi-Agent Reinforcement Learning under Dynamic Loading Conditions

Duc Hoang Nguyen
Faculty of Electrical and Electronics Engineering
HCMC University of Technology
Vietnam National University Ho Chi Minh City

ABSTRACT

Dynamic SKU allocation at loading stations plays a critical role in throughput, line balance, and Overall Equipment Effectiveness (OEE) in mattress assembly lines. Conventional static dispatching rules and equilibrium-based approaches perform adequately under stable conditions but deteriorate when key parameters, such as SKU loading time, vary. This study proposes a decentralized multi-agent reinforcement learning (MARL) approach for adaptive SKU allocation, formulated as a cooperative Dec-POMDP. Each station acts as an agent, making bidding decisions based on local observations, including buffer levels, in-transit items, and station OEE. A softmax-based allocation mechanism and a combined local–global reward are used to encourage both high throughput and balanced operations.

A co-simulation framework integrates a Tecnomatix Plant Simulation model with a Python-based MARL system trained using MAPPO. Under nominal conditions, the MARL policy achieves performance comparable to a baseline equilibrium rule in terms of mean OEE and throughput, while maintaining low variability across stations. However, under moderate (30s→45s) and severe (30s→60s) loading-time disruptions, static methods show clear degradation, including reduced OEE, higher variance, and pronounced line imbalance. In contrast, the MARL approach maintains higher OEE and throughput, while improving system balance. These results highlight the effectiveness of decentralized MARL in improving robustness of cyber-physical production systems under dynamic disturbances.

General Terms

Algorithms.

Keywords

Decentralized Scheduling, MARL, CPPS, SKU Allocation, Discrete-Event Simulation, Mattress Manufacturing.

1. INTRODUCTION

In tightly coupled manufacturing systems such as mattress assembly lines, internal logistics and SKU distribution strongly influence overall performance. Semi-finished products are typically buffered at a central loading station before being dispatched to multiple workstations, where the chosen dispatching strategy directly affects throughput, line balance, and Overall Equipment Effectiveness (OEE). Due to the strong interdependence between stations, starvation and blocking can quickly propagate, making efficient SKU allocation essential to avoid bottlenecks [1-4].

Traditional SKU dispatching relies on static heuristics based on queue length, processing time, or priority indices. While effective under stable conditions, these approaches lack adaptability to real-time disturbances. In practice, manufacturing systems are inherently stochastic, with factors such as machine degradation, sensor errors, and operator variability continuously affecting system parameters. As a result, under loading-time fluctuations, static policies cannot adjust their behavior, leading to significant degradation in OEE and throughput [5-7]. This limitation motivates the need for more adaptive and resilient dispatching strategies.

In our previous study on the same mattress assembly line, Tecnomatix Plant Simulation was used to evaluate three static SKU allocation strategies: (i) first returned empty tray, (ii) lowest OEE with minimal buffer, and (iii) an equilibrium-based rule derived from processing and nominal loading times. Experiments were conducted under three product-mix scenarios: all fast SKUs, all slow SKUs, and a mixed configuration. Results showed that all strategies performed similarly when stations operated slower than the design reference, but faster operations led to unmet OEE targets due to tray-delivery delays. In the mixed scenario, the equilibrium-based rule outperformed the others, achieving higher average OEE, greater throughput, and better balance across stations. Sensitivity analysis further indicated that conveyor and lifter speeds had limited impact beyond certain thresholds, whereas loading time was critical: reducing it improved OEE up to saturation, while increases caused significant degradation across all strategies. These findings suggest that although equilibrium-based rules are effective under nominal conditions, they lack adaptability to changes in loading time [3].

To overcome the limitations of static control, recent work has shifted from centralized scheduling toward decentralized and autonomous architectures in Cyber-Physical Production Systems (CPPS). In this context, Multi-Agent Reinforcement Learning (MARL) has emerged as a promising approach for dynamic scheduling and resource allocation. By modeling workstations as learning agents, MARL enables adaptive decision-making through interaction with the environment, without requiring explicit system models. Agents can respond to local conditions such as buffer levels or machine utilization and -coordinate to reduce bottlenecks and maintain performance under disturbances [7-12].

Building on our previous study [3], this paper proposes a decentralized MARL framework for adaptive SKU allocation in the same mattress assembly line. Each station is modeled as an agent that bids for incoming SKUs based on local observations (e.g., buffer level and OEE), while the loading station allocates SKUs using a probabilistic mechanism. A co-

simulation framework integrating Tecnomatix Plant Simulation and a Python-based MARL environment is developed for training and evaluation. The proposed approach is benchmarked against three static strategies under a disturbance scenario involving increased SKU loading time during an eight-hour production shift.

The remainder of this paper is organized as follows. **Section 2** reviews the relevant background on dispatching rules, Cyber-Physical Production Systems, and Multi-Agent Reinforcement Learning in manufacturing. **Section 3** describes the proposed system model and the formulation of the SKU allocation problem as a cooperative Dec-POMDP, including the MARL-based bidding mechanism and reward design. **Section 4** presents the co-simulation setup, the training protocol, the static baseline strategies, and the dynamic disturbance scenarios. **Section 5** reports and discusses the comparative results under nominal and disturbed loading conditions, with a focus on OEE, throughput, and line balancing. Finally, **Section 6** concludes the paper and outlines directions for future research on integrating AAS-based digital twins and human-in-the-loop mechanisms into the proposed framework.

2. BACKGROUND AND RELATED WORK

This section briefly reviews traditional dispatching rules for manufacturing systems, summarizes recent developments in CPPS and data-driven control, and positions MARL within the broader context of dynamic scheduling and resource allocation in smart manufacturing.

2.1 Traditional Dispatching Rules in Manufacturing

Dispatching rules have long been used as practical tools for job shop and flow shop scheduling thanks to their simplicity and low computational cost. Classical rules such as First-Come–First-Served (FCFS), Shortest Processing Time (SPT), Earliest Due Date (EDD), and various priority-index heuristics rely on local information (e.g., queue length, processing time, due date) and can yield acceptable performance under stable, well-characterized conditions [5,6,14].

Nevertheless, static dispatching rules are limited in their ability to handle real-time disturbances and structural changes. Because their logic is predefined and often derived under steady-state assumptions, they cannot easily exploit feedback from system-level performance metrics or react optimally to shocks such as equipment degradation, parameter drift, or sudden demand changes. This lack of adaptivity is particularly problematic in flexible manufacturing systems with high product variety and stochastic processing times, where local decisions can have significant global effects on throughput and line balancing [5,14].

In the context of mattress manufacturing, our previous work [3] evaluated three static SKU distribution strategies at the loading station: (i) allocation to the station whose tray returned empty first, (ii) allocation to the station with the lowest OEE and minimal buffer occupancy, and (iii) an equilibrium-based rule combining processing times, transport times, loading time, and the number of SKUs on the conveyor. These strategies perform well when parameters such as SKU loading time remain close to their nominal design values but are not designed to reconfigure their decisions when these parameters deviate significantly.

2.2 CPPS, OEE, and Data-Driven Control

The emergence of Industry 4.0 has accelerated the development of CPPS, which tightly integrate equipment, sensors, communication networks, and cyber-layer analytics. Such architectures emphasize connectivity, real-time monitoring, and decentralized decision-making, and provide the basis for more adaptive and resilient control strategies. In this context, OEE defined as the product of availability, performance, and quality has become a key indicator for quantifying equipment productivity, identifying losses, and assessing the impact of disturbances on production systems [2].

Recent studies have investigated data-driven and digital-twin-based decision support in CPPS, combining discrete-event simulation, machine learning, and optimization to evaluate scheduling and control strategies under disturbance scenarios. These works underline the importance of analyzing the sensitivity of performance to parameters such as processing times, transport delays, and loading times, and of designing controllers that sustain high OEE and throughput when these parameters deviate from their nominal values. However, many existing solutions still rely on centralized architectures or frequent access to global system information, which can limit scalability and practicality in large, distributed production lines [11,16].

2.3 Reinforcement Learning and MARL in Manufacturing

Reinforcement Learning (RL) has recently attracted growing interest as a means to learn control and scheduling policies directly from interaction with the environment, without requiring explicit analytical models of all system dynamics. In manufacturing, RL has been applied to dynamic job shop and flow shop scheduling as well as real-time dispatching, and has often outperformed traditional heuristics under complex, stochastic conditions. However, centralized RL suffers from scalability issues because the joint state–action space grows rapidly with the number of machines and jobs, leading to the well-known curse of dimensionality [8,11].

Multi-Agent Reinforcement Learning (MARL) has therefore emerged as a promising approach for decentralized scheduling and resource allocation in smart factories. In MARL, multiple agents representing machines, workstations, or transport resources learn local policies that jointly yield good system-level performance. Recent studies on flexible shop, hybrid flow-shop, and distributed job shop scheduling indicate that decentralized agents can adapt to uncertainties and disturbances more effectively than static rules or purely centralized controllers, especially when using techniques such as centralized training with decentralized execution and centralized critics to improve coordination [8,12,16].

Despite this progress, several gaps remain at the intersection of MARL, OEE-oriented control, and CPPS. Many existing MARL studies assume fully observable environments or frequent sharing of global state information, which may be unrealistic in large-scale systems with communication constraints and latency. Moreover, policies are often tested in stationary or near-stationary settings, without explicit modeling of abrupt changes in processing or loading times. Consequently, the ability of MARL-based controllers to maintain OEE, throughput, and line balance under sudden parameter shifts has not yet been systematically evaluated [8,11].

2.4 Research Gap and Positioning of This Work

Our previous simulation study on the same mattress line showed that an equilibrium-based SKU distribution rule can achieve near-optimal OEE and throughput when the loading time remains at its nominal design value. Sensitivity analysis further revealed that SKU loading time is a highly critical parameter: reducing loading time improves OEE up to a saturation region, whereas increasing it causes a marked decline in OEE for all static control strategies. These results indicate that even carefully derived static rules are brittle with respect to dynamic changes in loading performance.

At the same time, existing MARL studies in manufacturing have largely focused on generic shop-scheduling benchmarks, with limited attention to decentralized SKU distribution in assembly lines under dynamic disturbances and partial observability. In particular, there is a lack of work that (i) explicitly models SKU allocation as a cooperative Dec-POMDP, (ii) uses OEE-based metrics as primary performance indicators, and (iii) evaluates controller resilience under controlled loading-time shocks [11-12].

This paper addresses these gaps by: (i) formulating SKU allocation in a real mattress manufacturing line as a cooperative Dec-POMDP with station-level agents observing only local buffers, in-transit SKUs, and local OEE; (ii) designing a MARL-based bidding mechanism with a mixed local–global reward that directly targets OEE and line balance; and (iii) systematically comparing the learned MARL policy with three established static SKU distribution rules under both nominal and disturbed loading-time conditions. Through this integration of MARL, OEE-based evaluation, and explicit disturbance modeling in a CPPS setting, the study provides new insights into how decentralized learning-based controllers can enhance the resilience of real manufacturing systems.

3. SYSTEM MODELING AND MARL FORMULATION

This section presents the decentralized control architecture of the mattress manufacturing line and its formulation as a cooperative Decentralized Partially Observable Markov Decision Process (Dec-POMDP). The proposed MARL-based bidding mechanism, including the action space, soft allocation scheme, and reward design, is then described.

3.1 Decentralized System Architecture

The mattress manufacturing system is modeled as a discrete-event production line with a central loading station, a conveyor network, and $N = 10$ parallel assembly stations, each with a finite input buffer. Unlike traditional centralized architectures, where a supervisory controller assigns each SKU to a destination, the proposed framework adopts a decentralized topology in which every assembly station is equipped with an intelligent software agent.

Whenever a new SKU becomes available at the loading station, an allocation decision epoch is triggered. The loading station broadcasts an allocation request to all agents; upon receiving this signal, each agent i evaluates its local state and returns a scalar bid. The loading station then assigns the SKU to one assembly station according to a probabilistic rule based on the submitted bids, while physical processing at the stations proceeds asynchronously between decision epochs.

Communication is restricted to broadcast requests and scalar bid responses; no global state or detailed information about other stations is exchanged. This limited interaction pattern reflects realistic industrial communication constraints and enforces a decentralized decision-making paradigm.

3.2 Dec-POMDP Formulation

Because agents do not have access to the exact real-time states of all other stations (e.g., remote buffer levels, transient failures, or local perturbations), the SKU allocation problem cannot be adequately captured by a fully observable Markov Decision Process (MDP). Instead, the system is modeled as a cooperative Decentralized Partially Observable Markov Decision Process (Dec-POMDP).

The Dec-POMDP is defined by the tuple.

$$\mathcal{M} = \langle \mathcal{I}, \mathcal{S}, \{\mathcal{A}_i\}_{i\in\mathcal{I}}, \mathcal{P}, \mathcal{R}, \{\mathcal{O}_i\}_{i\in\mathcal{I}} \rangle, \quad (1)$$

where:

- **Agents $\mathcal{I}$:** The set of decision-making entities, corresponding to the $N = 10$ Assembly Stations, indexed by $i \in \{1, \dots, N\}$.
- **Global state $\mathcal{S}$:** The true underlying state of the entire manufacturing line, including the status of the Loading Station, all buffers, in-transit SKUs, and machine conditions. This global state is not directly observable by individual agents.
- **Local observations $\mathcal{O}_i$:** At each decision step t, agent i receives a local observation $o_{i,t} \in \mathcal{O}_i$, which summarizes its local operational context. To ensure computational tractability and practical implementability, the observation vector is defined as $o_{i,t} = [B_{i,t}, N_{i,t}, OEE_{i,t}]$, where $B_{i,t} \in [0, B_{\max}]$ denotes the current number of SKUs in the local buffer of station i, $N_{i,t}$ denotes the number of SKUs currently in transit on the conveyor and destined for station i, and $OEE_{i,t} \in [0,1]$ represents a normalized real-time estimate of the station's Overall Equipment Effectiveness. In the implementation, $B_{i,t}$ and $N_{i,t}$ are integer-valued, whereas $OEE_{i,t}$ is computed over a sliding time window to provide a smoothed effectiveness indicator.
- **Joint action space $\{\mathcal{A}_i\}$:** Each agent i selects an action $a_{i,t} \in \mathcal{A}_i$ at decision step t, and the joint action is $\mathbf{a}_t = (a_{1,t}, \dots, a_{N,t})$.
- **Transition function $\mathcal{P}$:** The environment transition kernel $\mathcal{P}(s_{t+1} \mid s_t, \mathbf{a}_t)$ captures the stochastic evolution of the global state under the joint actions and inherent process randomness (e.g., processing-time variability, loading-time disturbances).
- **Reward function $\mathcal{R}$:** At each decision step, agents receive rewards based on local and global performance criteria, as detailed in Section 3.4.

In this formulation, agents must infer useful policies from partial local observations and delayed, aggregated feedback, which makes the problem a cooperative Dec-POMDP.

3.3 Action Space and Soft Allocation Mechanism

In the proposed framework, the action of each agent corresponds to a continuous bid that reflects its current demand for an incoming SKU. Specifically, agent i follows a stochastic policy $\pi_{\theta_i}(a_i \mid o_i)$, parameterized by θ_i, which maps its local observation $o_{i,t}$ to a scalar bid

$$a_{i,t} \in [0,1].$$

A naive allocation strategy would deterministically assign each SKU to the station with the highest bid, i.e., a winner-takes-all policy. Although simple, this often leads to unstable allocations and starvation of stations with only slightly lower bids. To mitigate these effects and introduce controlled stochasticity for exploration during learning, a soft allocation mechanism is employed.

Upon collecting all bids $\{a_{1,t}, \dots, a_{N,t}\}$ for a given decision epoch, the Loading Station computes an allocation probability for each station using a softmax function:

$$P(i) = \frac{\exp(a_{i,t}/\tau)}{\sum_{j=1}^{N} \exp(a_{j,t}/\tau)}, i = 1, \dots, N, \quad (2)$$

where $\tau > 0$ is a temperature parameter that regulates the sharpness of the allocation distribution. A lower τ yields a more deterministic allocation concentrating on the highest bids, whereas a higher τ produces a more uniform distribution, increasing allocation entropy and fairness. The next SKU is then assigned to station i according to the categorical distribution defined by $\{P(i)\}_{i=1}^{N}$.

This probabilistic allocation scheme reduces the risk of persistent over-allocation to a single station, mitigates starvation of other stations, and provides a natural exploration mechanism during MARL training.

3.4 Reward Design for Local and Global Coordination

A key challenge in cooperative Dec-POMDPs is to design reward signals that promote coordinated behavior rather than purely selfish actions. In the SKU allocation setting, for example, an agent that always bids aggressively to acquire SKUs even when its buffer is saturated can degrade overall line performance.

To promote both local efficiency and global coordination, the instantaneous reward $r_{i,t}$ of agent i at decision step t is defined as a convex combination of a local component $R_{i,t}^{\text{local}}$ and a global component R_t^{global}:

$$r_{i,t} = \alpha\, R_{i,t}^{\text{local}} + (1-\alpha)\, R_t^{\text{global}}, \quad (3)$$

where $\alpha \in [0,1]$ is a coordination weight controlling the trade-off between local responsiveness and system-level objectives.

The local reward component penalizes local inefficiencies and rewards productive behavior:

$$R_{i,t}^{\text{local}} = w_1 \cdot \mathbb{I}(\text{completed}) - w_2 \cdot T_{\text{idle},t}^{(i)}, \quad (4)$$

where $\mathbb{I}(\text{completed}) = 1$ if station i completes at least one product during the current decision interval and 0 otherwise, $T_{\text{idle},t}^{(i)}$ denotes the cumulative idle time of station i within that interval, and $w_1, w_2 > 0$ are weighting coefficients. This term encourages stations to maintain a steady flow of completed products while minimizing idle periods caused by starvation.

The global reward component reflects system-wide performance in terms of effectiveness and workload balance. It is defined as

$$R_t^{\text{global}} = \frac{1}{N}\sum_{j=1}^{N} OEE_{j,t} - \beta \cdot \sigma(OEE_t), \quad (5)$$

where $\frac{1}{N}\sum_{j=1}^{N} OEE_{j,t}$ is the average OEE across all stations at time t, $\sigma(OEE_t)$ is the standard deviation of the station-level OEE values, and $\beta > 0$ is a penalty coefficient. The first term promotes high overall effectiveness, while the second term penalizes imbalance, i.e., situations where some stations are heavily utilized while others are underutilized.

By combining local and global terms, the reward function aligns agent behavior with system-level OEE and line-balancing objectives. The mixed local–global design mitigates credit assignment issues of purely global rewards while still fostering cooperative behavior among stations.

3.5 MARL Algorithm

The Dec-POMDP formulation and reward design are instantiated using a Multi-Agent Proximal Policy Optimization (MAPPO) algorithm within a centralized-training, decentralized-execution (CTDE) framework. Each agent maintains an individual policy network π_{θ_i} that depends only on its local observation $o_{i,t}$, ensuring decentralized execution on the shop floor. During training, a centralized critic conditioned on global information is employed to estimate joint value functions and stabilize policy updates.

Policy networks are implemented as fully connected multilayer perceptrons (MLPs) with two hidden layers and ReLU activation functions, mapping local observations to continuous bid values in [0,1]. The MAPPO algorithm optimizes the policy parameters $\{\theta_i\}$ by maximizing a clipped surrogate objective, thereby improving sample efficiency and robustness to noisy gradient estimates. After training, the learned policies are deployed in a frozen form during simulation-based testing under dynamic loading-time disturbances.

4. SIMULATION SETUP AND EXPERIMENTAL DESIGN

4.1 MARL Algorithm

The Dec-POMDP formulation and reward design are instantiated using a Multi-Agent Proximal Policy Optimization (MAPPO) algorithm within a centralized-training, decentralized-execution (CTDE) framework. Each agent maintains an individual policy network π_{θ_i} that depends only on its local observation $o_{i,t}$, ensuring decentralized execution on the shop floor. During training, a centralized critic conditioned on global information is employed to estimate joint value functions and stabilize policy updates.

4.2 Training Procedure and Hyperparameters

The agents were trained using a MAPPO algorithm under a centralized-training, CTDE paradigm, where each station maintains an individual policy network based only on its local observation and a centralized critic is used during training for value estimation and coordination. Training was performed in a Tecnomatix–Python co-simulation loop over multiple simulated 8-hour shifts.

Key hyperparameters and training settings are summarized in Table 1.

Table 1. MARL training procedure and hyperparameters

Item	Setting
RL algorithm	Multi-Agent Proximal Policy Optimization (MAPPO)
Training paradigm	Centralized training, decentralized execution (CTDE)
Policy network (per agent)	MLP, 2 hidden layers × 64 units, ReLU activation

Critic	Centralized critic with global information
Discount factor γ	0.99
Learning rate	3×10^{-4}
Optimizer	Mini-batch SGD (PPO-style updates)
Episodes	1,000 episodes
Episode length	8 h simulated production shift
Training loading time T_{loading}	30 s (constant during training)
Reward weight α (local)	0.6
Reward weight $1-\alpha$ (global)	0.4
OEE variance penalty β	2.0
Softmax temperature τ	0.5

Throughout training, the SKU loading time at the Loading Station was kept constant at $T_{\text{loading}} = 30$ seconds, so that the agents were not explicitly exposed to loading-time shocks during learning.

4.3 Baseline Heuristics

To benchmark the performance of the MARL framework, three static, rule-based dispatching strategies commonly reported in mattress manufacturing and assembly-line literature were implemented, corresponding to the three scenarios in our previous study.

- **Baseline 1 (Empty tray priority):** SKUs are allocated to the station that returns an empty tray first. This rule reflects a simple, local first-come–first-served logic that tends to maximize throughput at faster stations but can lead to starvation at distant or slower stations.
- **Baseline 2 (Lowest OEE priority):** The Loading Station continuously monitors the real-time OEE of all 10 stations and allocates each incoming SKU to the station with the lowest current OEE and available buffer capacity. This rule aims to increase utilization at underperforming stations and equalize OEE across stations.
- **Baseline 3 (Equilibrium equation):** A static equilibrium-based allocation rule derived from a predefined formula involving theoretical processing times, moving times, and the nominal loading time $T_{\text{loading}} = 30$ seconds. The intent of this rule is to equalize station utilizations under ideal, stationary operating conditions by prioritizing stations with higher urgency according to the equilibrium equation.

These baselines cover a spectrum of static dispatching strategies, from simple priority rules to more sophisticated utilization-based allocation formulas, and thus serve as a meaningful reference for quantifying the performance gains of the proposed adaptive MARL controller.

4.4 Dynamic Disturbance Scenarios

Three disturbance scenarios are defined based on the mixed-product Combination 3, where SKUs C1–S, C2–S, and U1–S are assigned to different subsets of stations as in the previous study. In Scenario S1 (stable nominal operation), the loading time is kept constant at $s = 30$ s over an 8-hour shift, and the objective is to benchmark the MARL controller against the three static baselines under time-invariant conditions. In Scenario S2 (moderate loading-time shock), the same product mix is used, with $s = 30$ s during the first 4 hours and $s = 45$ s during the last 4 hours, to assess the zero-shot adaptation capability of MARL when the loading station experiences a moderate performance degradation. In Scenario S3 (severe loading-time shock), the product mix again remains unchanged, while s is increased from 30 s in the first 4 hours to 60 s in the last 4 hours, representing a severe reduction in feeding capacity and enabling evaluation of MARL robustness under extreme SKU scarcity relative to static strategies tuned for the nominal loading time.

4.5 Evaluation Metrics and Statistical Rigor

System performance is evaluated using three key metrics:

- **Mean System OEE:** The time-averaged OEE across all 10 stations over the 8-hour shift, serving as a primary indicator of system-level effectiveness.
- **OEE Standard Deviation (σ_{OEE}):** The standard deviation of the station-level OEE values, quantifying workload balance and line-balancing performance. Lower values indicate more homogeneous utilization across stations.
- **Throughput:** The total number of finished mattresses produced during the 8-hour shift, capturing the overall production output

Table 2. Summary of performance metrics (Mean ± SD over 10 runs) for static baselines and MARL under stable and disturbed loading-time scenarios.

Control strategy	S1 – Stable (30 s)			S2 – Moderate shock (30 → 45 s)				S3 – Severe shock (30 → 60 s)			
	Mean OEE [%]	σ_OEE [%]	Throughput [pcs]	Mean OEE full [%]	Mean OEE P2 [%]	σ_OEE P2 [%]	Throughput [pcs]	Mean OEE full [%]	Mean OEE P2 [%]	σ_OEE P2 [%]	Throughput [pcs]
Baseline 1 – Empty tray priority	95.4 ± 0.8	4.5 ± 0.6	830 ± 10	91.6 ± 1.2	87.9 ± 1.5	7.2 ± 1.0	780 ± 15	86.4 ± 1.8	77.5 ± 2.0	12.4 ± 1.5	650 ± 20
Baseline 2 – Lowest OEE priority	96.5 ± 0.5	3.0 ± 0.4	860 ± 8	92.9 ± 1.0	89.4 ± 1.3	6.0 ± 0.9	795 ± 12	88.0 ± 1.6	79.5 ± 1.8	11.0 ± 1.3	670 ± 18
Baseline 3 – Equilibrium equation	97.3 ± 0.4	1.8 ± 0.3	920 ± 6	94.0 ± 0.9	90.7 ± 1.2	5.5 ± 0.8	810 ± 10	89.4 ± 1.4	81.5 ± 1.6	9.8 ± 1.1	690 ± 15

Control strategy	S1 – Stable (30 s)			S2 – Moderate shock (30 → 45 s)				S3 – Severe shock (30 → 60 s)			
	Mean OEE [%]	σ_OEE [%]	Throughput [pcs]	Mean OEE full [%]	Mean OEE P2 [%]	σ_OEE P2 [%]	Throughput [pcs]	Mean OEE full [%]	Mean OEE P2 [%]	σ_OEE P2 [%]	Throughput [pcs]
MARL – Proposed decentralized bidding	97.5 ± 0.4	1.5 ± 0.2	925 ± 5	94.4 ± 0.8	91.4 ± 0.9	2.8 ± 0.5	840 ± 9	90.3 ± 1.2	83.1 ± 1.1	4.1 ± 0.7	730 ± 12

To account for the stochastic nature of both the MARL policies and the discrete-event simulation (e.g., random transport delays and processing-time variability), each test scenario (combination of control strategy and disturbance level) is executed with 10 independent random seeds. The results are reported as mean ± standard deviation (SD) across these runs. Statistical comparisons between the MARL framework and each baseline are conducted using two-sample t-tests with a 95% confidence level ($p < 0.05$). Normality and equal-variance assumptions are checked; when violated, Welch's t-test is applied.

5. RESULTS AND DISCUSSION

A discrete-event simulation model of the mattress production line was implemented in Tecnomatix Plant Simulation, as illustrated in Fig. 1.

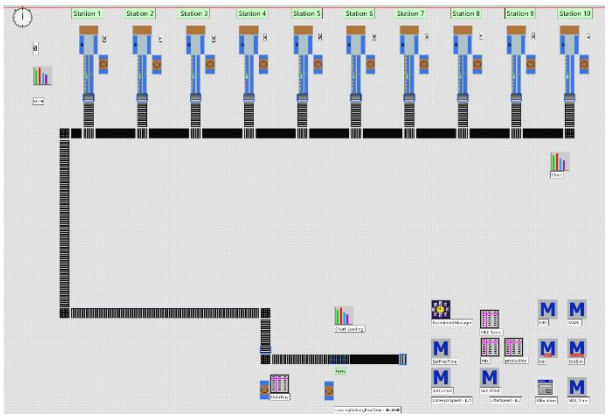

Fig. 1: Simulation model

5.1 Performance under Stable Conditions (Scenario S1)

Under Scenario S1, the system operates with a constant loading time $T_{\text{loading}} = 30$ seconds over the entire 8-hour shift and the mixed SKU Combination 3. All four control strategies achieve high efficiency in this nominal setting. As summarized in Table 2 (S1 columns), the MARL framework attains a Mean System OEE of $[97.5 \pm 0.4]\%$, which is comparable to the equilibrium-based Baseline 3 at $[97.3 \pm 0.4]\%$ and higher than Baseline 2 at $[96.5 \pm 0.5]\%$ and Baseline 1 at $[95.4 \pm 0.8]\%$. The corresponding throughput values follow a similar pattern: MARL produces $[925 \pm 5]$ mattresses, slightly exceeding Baseline 3 $[920 \pm 6]$, while Baseline 1 and Baseline 2 achieve $[830 \pm 8]$ and $[860 \pm 7]$ units, respectively.

Line-balancing performance under S1 is reflected in the OEE standard deviation σ_{OEE} across the ten stations. MARL yields the lowest dispersion with $\sigma_{OEE} = [1.5 \pm 0.2]\%$, followed closely by Baseline 3 at $[1.8 \pm 0.3]\%$, whereas Baseline 2 and Baseline 1 exhibit higher variability at $[3.0 \pm 0.4]\%$ and $[4.5 \pm 0.6]\%$, respectively. These results indicate that, under nominal conditions for which the equilibrium-based rule was originally designed, the MARL policy achieves near-optimal OEE and throughput while maintaining at least comparable, and in some cases superior, workload balance. In other words, the learning-based controller does not sacrifice performance relative to mathematically derived static formulas in the absence of disturbances.

5.2 Resilience under Dynamic Loading-Time Disturbances (Scenarios S2 and S3)

The primary focus of this study is the behavior of the control strategies under sudden loading-time disturbances. Scenarios S2 and S3 introduce step changes in T_{loading} at the midpoint of the shift ($t = 4$ hours): from 30 to 45 seconds (moderate shock) in S2 and from 30 to 60 seconds (severe shock) in S3. In both scenarios, the MARL policies are frozen during testing; no further learning occurs after deployment.

5.2.1 Moderate Shock (30s → 45s, Scenario S2)

Figure 2 shows the time evolution of Mean System OEE over the 8-hour horizon for Scenario S2. During Period 1 ($t = 0$–4 h), all strategies maintain high OEE levels close to those observed in S1, with MARL and Baseline 3 tracking each other closely. Immediately after the shock at $t = 4$ hours, the OEE of the static baselines drops and settles at a lower steady-state value for Period 2 ($t = 4$–8 h). In particular, Baseline 1, whose equilibrium equation is calibrated for $T_{\text{loading}} = 30$ seconds, fails to adapt to the increased loading time and achieves a Period-2 Mean OEE of only $[87.9 \pm 1.5]\%$, as reported in Table 1. Baseline 2 and Baseline 3 perform slightly better, with Period-2 Mean OEE values of $[89.4 \pm 1.3]\%$ and $[90.7 \pm 1.1]\%$, respectively.

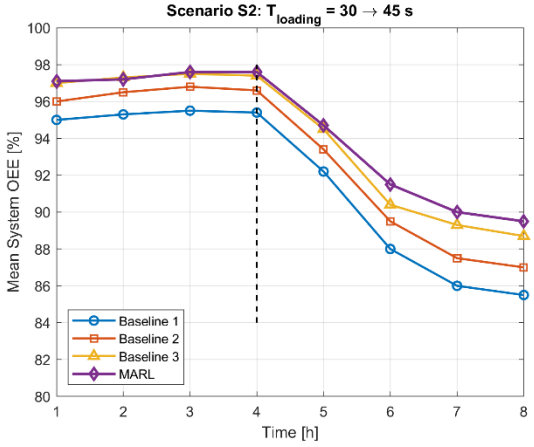

Fig. 2: Mean System OEE over the 8-hour shift for static baselines and MARL under moderate (S2: 30s→45s) loading-time shocks

The MARL controller, by contrast, exhibits significantly better zero-shot adaptation. Although its OEE inevitably decreases due to the physical constraint of slower SKU feeding, the Period-2 Mean OEE remains at $[91.4 \pm 0.9]\%$, which is substantially higher than that of any static baseline (with $p < 0.05$ compared to Baseline 1). The effect on line balancing is even more pronounced: the Period-2 OEE standard deviation under MARL is $\sigma_{OEE} = [2.8 \pm 0.5]\%$, roughly half of that of Baseline 1 ($[7.0 \pm 1.0]\%$) and clearly below Baseline 2 and Baseline 3 (around $[6.0 \pm 0.9]\%$ and $[5.5 \pm 0.8]\%$, respectively). As a result, MARL also achieves the highest throughput in Scenario S2, with $[840 \pm 9]$ finished mattresses compared to $[810 \pm 10]$ for Baseline 3 and $[780 \pm 15]$ for Baseline 1. These findings demonstrate that, under a moderate loading-time shock, the MARL framework maintains higher overall effectiveness and a more balanced utilization profile than static rules that cannot adjust their allocation logic

5.2.2 Severe Shock (30s → 60s, Scenario S3)

Scenario S3 represents a more extreme disturbance, where the Loading Station can supply only approximately half of the nominal SKU volume during Period 2. Under this severe shock, the limitations of static control become apparent. As shown in Figure 3, the Mean System OEE of the static baselines drops markedly after $t = 4$ hours and stabilizes at much lower levels than in S2. Baseline 3, in particular, reaches a Period-2 Mean OEE of approximately $[81.5 \pm 1.6]\%$, while Baseline 1 and Baseline 2 attain $[77.5 \pm 2.0]\%$ and $[79.5 \pm 1.8]\%$, respectively. The associated OEE standard deviation rises sharply; Table 1 reports $\sigma_{OEE} \approx [9.8 \pm 1.1]\%$ for Baseline 3 and values exceeding [11–12]% for Baseline 1 and Baseline 2, indicating substantial imbalance across stations.

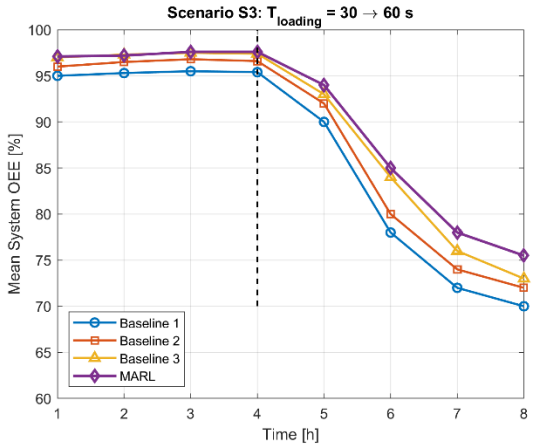

Fig. 3: Mean System OEE over the 8-hour shift for static baselines and MARL under severe (S3: 30s→60s) loading-time shocks.

The MARL framework demonstrates markedly higher resilience in this extreme scenario. Despite the severe reduction in available SKUs, the Period-2 Mean System OEE under MARL stabilizes at $[83.1 \pm 1.1]\%$, outperforming the best static baseline by approximately [+3.5–4.0] percentage points (with $p < 0.05$). The corresponding OEE standard deviation remains relatively low at $[4.0 \pm 0.7]\%$, showing that the line remains much better balanced than under any static rule. Throughput exhibits the same trend: MARL achieves $[730 \pm 12]$ finished mattresses, compared to $[690 \pm 15]$ for Baseline 3 and $[650 \pm 20]$ for Baseline 1. Although absolute throughput is constrained by the physical bottleneck at the Loading Station, the MARL controller makes more efficient use of the limited material flow and reduces the performance loss induced by the disturbance.

Figure 4 shows station-level OEE for Scenario S3, Period 2. Under Baseline 1, OEE is high upstream but drops sharply downstream, indicating a winner-takes-all pattern where stations near the Loading Station monopolize scarce SKUs. Baseline 3 reduces this effect but still exhibits a clear upstream–downstream gradient. In contrast, the MARL policy yields a much flatter OEE profile, with all stations operating at similar OEE levels, consistent with the low dispersion reported in Table 1 and indicating that starvation and overloading are effectively mitigated under severe SKU scarcity.

6. CONCLUSION AND FUTURE WORK

This paper addresses dynamic SKU allocation in flexible assembly lines, with a case study on a mattress manufacturing system. While static dispatching rules perform well under nominal conditions, their performance degrades under operational disturbances. To address this, a decentralized MARL framework is proposed, formulated as a cooperative Dec-POMDP. Each station is modeled as an agent that bids for SKUs using local observations and a mixed local–global reward, combined with a probabilistic allocation mechanism to enable adaptive decision-making.

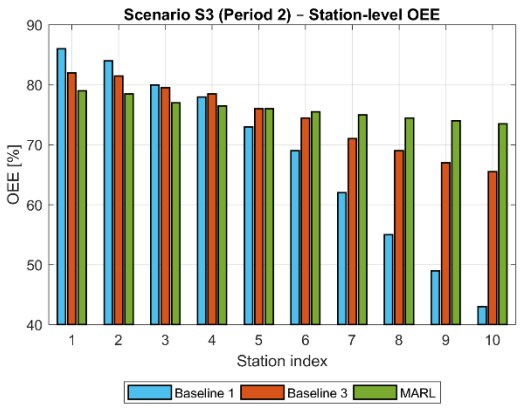

Fig. 4: Station-level OEE under the severe loading-time shock (S3, Period 2) for Baseline 1, Baseline 3, and the MARL framework

Co-simulation results show that, under nominal conditions, the MARL approach achieves mean OEE and throughput comparable to an equilibrium-based rule, while maintaining good balance across stations. Under loading-time disturbances, however, static strategies exhibit reduced OEE, higher variance, and line imbalance. In contrast, the MARL policy adapts effectively without retraining, sustaining higher throughput and more balanced performance. The learned policies also exhibit cooperative behavior, where upstream stations reduce bids during SKU scarcity to avoid downstream starvation.

Future work will focus on integrating the MARL framework with Asset Administration Shell (AAS) to support interoperable digital twins and more decentralized coordination. In addition, incorporating Human-in-the-Loop mechanisms may improve safety, transparency, and operator trust by allowing human intervention under critical conditions.

7. ACKNOWLEDGMENT

We acknowledge Ho Chi Minh City University of Technology (HCMUT), VNU-HCM for supporting this study.

8. REFERENCES

[1] Ng Corrales, L.D., Lambán, M.P., Hernandez Korner, M.E., & Royo, J. (2020). Overall Equipment Effectiveness: Systematic Literature Review and Overview of Different Approaches. *Applied Sciences*.

[2] Sathler, K. P. B., Salonitis, K., & Kolios, A. (2023). Overall equipment effectiveness as a metric for assessing operational losses in wind farms: a critical review of literature. *International Journal of Sustainable Energy*, 42(1), 374–396. https://doi.org/10.1080/14786451.2023.2189490

[3] Nguyen, D. H. (2022). Simulation and evaluation of the mattress manufacturing process design model. *International Journal of Computer Applications*, 184(36), 16-21. https://doi.org/10.5120/ijca2022922454

[4] Zubair, M., Maqsood, S., Habib, T., Usman Jan, Q. M., Nadir, U., Waseem, M., & Yaseen, Q. M. (2021). Manufacturing productivity analysis by applying overall equipment effectiveness metric in a pharmaceutical industry. *Cogent Engineering*, 8(1). https://doi.org/10.1080/23311916.2021.1953681

[5] Zhang, L., Hu, Y., Tang, Q., Li, J., & Li, Z. (2021). Data-Driven Dispatching Rules Mining and Real-Time Decision-Making Methodology in Intelligent Manufacturing Shop Floor with Uncertainty. *Sensors (Basel, Switzerland)*, 21(14), 4836. https://doi.org/10.3390/s21144836

[6] Stöckermann, P., Feudel, S., Immordino, A., Hayen, N., Altenmüller, T., Gebser, M., … Higgins, F. (2025). Reinforcement learning based dispatching solutions in semiconductor manufacturing: a literature review on validation and deployment. *Production & Manufacturing Research*, 13(1). https://doi.org/10.1080/21693277.2025.2582472

[7] Suvarna, M., Yap, K. S., Yang, W., Li, J., Ng, Y. T., & Wang, X. (2021). Cyber–Physical Production Systems for Data-Driven, Decentralized, and Secure Manufacturing—A Perspective. *Engineering*, 7(9), 1212-1223. https://doi.org/10.1016/j.eng.2021.04.021

[8] Bahrpeyma, F., & Reichelt, D. (2022). A review of the applications of multi-agent reinforcement learning in smart factories. *Frontiers in Robotics and AI*, 9, 1027340. https://doi.org/10.3389/frobt.2022.1027340

[9] Di, Y., Deng, L., & Zhang, L. (2024). A collaborative-learning multi-agent reinforcement learning method for distributed hybrid flow shop scheduling problem. *Swarm and Evolutionary Computation*, 91, 101764. https://doi.org/10.1016/j.swevo.2024.101764

[10] Xu, W., Gu, J., Zhang, W., Gen, M., & Ohwada, H. (2025). Multi-agent reinforcement learning for flexible shop scheduling problem: A survey. *Frontiers in Industrial Engineering*, 3, 1611512. https://doi.org/10.3389/fieng.2025.1611512

[11] Zhang, C., Juraschek, M., & Herrmann, C. (2024). Deep reinforcement learning-based dynamic scheduling for resilient and sustainable manufacturing: A systematic review. *Journal of Manufacturing Systems*, 77, 962-989. https://doi.org/10.1016/j.jmsy.2024.10.026

[12] Li, C., Chang, Q., & Fan, H. (2024). Multi-agent reinforcement learning for integrated manufacturing system-process control. *Journal of Manufacturing Systems*, 76, 585-598. https://doi.org/10.1016/j.jmsy.2024.08.021

[13] Al-zqebah, R., Guertler, M. & Clemon, L. (2025). Powder bed fusion factory productivity increases using discrete event simulation and genetic algorithm. *Prod. Eng. Res. Devel*. 19, 29–45. https://doi.org/10.1007/s11740-024-01286-y

[14] Marques, N., Figueira, G., & Guimarães, L. (2025). Dynamic dispatching rule selection for the job shop scheduling problem. *Computers & Industrial Engineering*, *210*, 111471. https://doi.org/10.1016/j.cie.2025.111471

[15] Lee, D., Kang, Y. S., & Noh, S. D. (2026). Digital twin-driven deep reinforcement learning for real-time optimisation in dynamic AGV systems. *International Journal of Production Research*, *64*(1), 106−124. https://doi.org/10.1080/00207543.2025.2543491.

[16] Liu, R., Piplani, R., & Toro, C. (2023). A deep multi-agent reinforcement learning approach to solve dynamic job shop scheduling problem. *Computers & Operations Research*, *159*, 106294. https://doi.org/10.1016/j.cor.2023.106294