Multi-Agent Simulation of Manufacturing Networks: Autonomous and Adaptive Decision-Making for Industry 4.0

Alysson Pereira e Pereira
Federal University of Maranhao
Av. dos Portugueses, 1966, Bacanga, São Luís/MA, Brazil.
alyssonpereira41@gmail.com

Frederic Menezes Ferreira
Federal University of Maranhao
Av. dos Portugueses, 1966, Bacanga, São Luís/MA, Brazil.
frederic.menezes@ufma.br

Alexandre Cesar Muniz de Oliveira
Federal University of Maranhao
Av. dos Portugueses, 1966, Bacanga, São Luís/MA, Brazil.
alexandre.cesar@ufma.br

RESUMO

Este trabalho aborda a modelagem e simulação de Sistemas Flexíveis de Manufatura (FMS) para otimizar a produção na Indústria 4.0. A pesquisa integra agentes inteligentes e aprendizado por reforço para aprimorar eficiência e flexibilidade na tomada de decisão de unidades autônomas. Propõe um framework multicritério que equilibra lucro, sustentabilidade e variabilidade, permitindo adaptação em tempo real. O ambiente multiagente facilita a comunicação entre unidades, redistribuindo tarefas para reduzir rejeições e aumentar a produtividade. A validação ocorreu por meio de simulações em redes com seis unidades fabris, cada uma monitorada por um agente que decide aceitar demandas conforme sua política, recursos e métricas de desempenho. Os resultados mostram que o modelo atinge os objetivos e equilibra critérios econômicos, de sustentabilidade e de customização.

PALAVRAS CHAVE. Sistemas de Manufatura Flexível; Sistemas Multiagentes; Aprendizado por Reforço.

ABSTRACT

This work addresses the modeling and simulation of Flexible Manufacturing Systems (FMS) to optimize industrial production in Industry 4.0. The research integrates intelligent agents and reinforcement learning to improve efficiency and flexibility in decision-making in autonomous units. It proposes a multi-criteria framework that balances profit, sustainability, and variability, allowing real-time adaptation. The multi-agent environment facilitates unit communication, redistributing tasks to reduce rejects and increase productivity. The validation was carried out by computational simulations of a small manufacturing network comprising six manufacturing units. Each unit has a dedicated agent to decide the demand acceptance according to its policy, resource, and performance measurements. The simulation results have shown that the decision model can pursue the target and equilibrate the economic, sustainability, and customization criteria. The study highlights the potential of autonomous and connected manufacturing networks to optimize industrial processes, promoting sustainability and competitiveness in Industry 4.0.

KEYWORDS. Flexible Manufacturing; Multi-agent Systems; Reinforcement Learning.

1. Introduction

Self-Organizing Manufacturing Networks (SOMN) are production systems that combine the Internet of Things (IoT) and artificial intelligence to operate autonomously and adaptively. In these systems, machines, devices, and software connect dynamically, possessing the ability to self-configure, self-optimize, and self-heal [Lu et al., 2021]. SOMN represents a shift from centralized manufacturing paradigms toward more dynamic and adaptive production environments, capable of producing more sustainable and customer-tailored products.

In this context, approaches based on multi-agent systems and reinforcement learning have proven promising, as evidenced in the works of [Zhang et al., 2022], which propose distributed architectures for intelligent decision-making in dynamic environments. Additionally, studies by [Wang et al., 2021] explore simulation and real-time scheduling using optimization techniques combined with learning, highlighting the feasibility of hybrid solutions for complex production environments. [Ferreira et al., 2023] introduces an initial framework for self-optimizing manufacturing networks using RL, a conceptual basis expanded in this work for a multi-agent environment.

The use of modern reinforcement learning algorithms, such as Proximal Policy Optimization (PPO), proposed by [Schulman et al., 2017a], has stood out for its stability and efficiency, being widely adopted in industrial scenarios, as seen in the studies of [Zhou et al., 2021]. To contextualize our use of RLlib/PPO, [Li et al., 2023] provides a comprehensive review of Deep Reinforcement Learning in smart manufacturing. Furthermore, [Popper et al., 2023] applies PPO in a multi-agent context for dynamic production facility planning, balancing economic and sustainability variables. This work incorporates such advances through the RLlib library [Liang et al., 2018] and the PettingZoo framework [Williams et al., 2021], offering a robust structure for distributed simulation with multiple agents. Additionally, multi-criteria aspects such as sustainability and customization have been discussed in recent studies, such as [Zhang and Gao, 2020] and [Váncza et al., 2019], pointing to the need for models that reconcile economic and socio-environmental objectives.

The proposal presented in this paper explores integrating an environment with multiple agents, which is adaptive to uncertainty that may occur in industrial scenarios and capable of handling distinct priorities (economic, sustainability, variability), promoting an adaptive global real-time balancing. The Economic Criterion focuses on maximizing profit or minimizing cost, considering factors like production efficiency, resource utilization, and operational expenses. Sustainability seeks to reduce environmental impact by minimizing energy consumption, emissions, and waste, promoting long-term ecological balance. Personalization or variability (local customer service) criterion involves adapting products or services to meet individual customer preferences or specifications, enhancing satisfaction and market competitiveness [Abualigah et al., 2023; Dennison et al., 2024; Cheng et al., 2023]. This work extends the manufacturing environment model initially proposed in [Ferreira et al., 2023], which considers a single peripheral unit. Considering dynamic processing priorities and capacities, we propose a multi-agent extension to reinsert demands into the networked system (multi-unit), minimizing the number of demand rejections.

This paper is organized as follows: Section 2 presents the framework of the environment, Section 3 includes the modeling and evaluation functions of the environment. Section 4 discusses the computational results, and Section 5 highlights the main findings and future research.

2. SOMN background

This section describes the Self-Optimised Manufacturing Network (SOMN) foundations, including the environment simulator, single decision-making agent modelling, the system's state variables, and the reinforcement learning techniques employed in this work [Ferreira et al., 2023; Lu et al., 2021; Kim et al., 2020].

We assume an environment composed of autonomous manufacturing units negotiating raw material quantities and lead times to meet the forecasted demand for customized products. Each unit contributes with its productive activity to the global multi-criteria objective involving costs, profit, and sustainability indicators. Concerns with variability are introduced through the aggregation of specific and regional characteristics to the structural ones.

2.1. Environment Variables

Stock balance $\overline{BA}$, for M global and local raw materials stored at the local unit used to feature each demand. The stock balance introduces uncertainty and delays to production. The inventory includes not only the raw materials physically present but also those that are incoming $\overline{IN}$ (already ordered but not received) and those already allocated to production $\overline{OU}$. Availability $\overline{AV}$ is calculated as:

$$\overline{AV} = \overline{BA} + \overline{IN} - \overline{OU} \tag{1}$$

Product requests (demands d) are received at specific time intervals and grouped by customer. The date/time records are the date of input DI_d and delivery deadline (date of output - DO_d), which must encompass the expected delivery lead time LT_d. Each request set d has an expected cost CO_d, space demanded if held temporarily in the yard SP_d, net profit per unit PR_d, and an eventual penalty PE_d for accepting but failing to deliver the request.

Variables ST_d and $\overline{FT_d}$ refer to each demand's request status and resource matrix. The cost CO_d, the space SP_d proportionally depends on $CO_d = AM_d \cdot \overline{FT_d} \cdot \overline{EU}$, where *overline variables* $\overline{*}$ represent an array $M-$dimensional, where M is the maximum number of raw material or features in the system and $\overline{EU}$ is expected unit price of each raw material. Other demand attributes are calculated using specific functions as shown in Equations 2 to 4.

$$LT_d = \tau(AM_d \cdot FT_d) \tag{2}$$

$$VA_d = \nu(FT_d) \tag{3}$$

$$SU_d = \sigma(FT_d^{-1}) \tag{4}$$

where functions τ, ν and σ represents abstractions over criteria calculations and *AM, VA, SU* represents amount, variability, sustainability respectively. A demand batch may assume different statuses ST_d throughout the production process, ranging from waste (-1), received (0), ready (1), rejected (2), produced (3), stored (4), to delivered (5) [Ferreira et al., 2023].

2.2. Reinforcement learning-based agents

Reinforcement learning algorithms can be categorized into on-policy and off-policy. On-policy methods directly optimize the current policy, resulting in significant stability but requiring more data samples. In dynamic environments, on-policy is preferable, while in scenarios with limited data collection, off-policy is advantageous. Both methods aim to optimize the agent's policy to maximize the accumulated reward, using interactions with the environment to collect data and improve decisions [Schulman et al., 2017b].

RL policy-gradient methods are both Proximal Policy Optimization (PPO) and Recurrent Proximal Policy Optimization (Recurrent PPO). The key difference is that Recurrent PPO incorporates Long Short-Term Memory (LSTM) networks to leverage information from past states. In this work, the Recurrent Proximal Policy Optimization (Recurrent PPO) is employed, not relying on an explicit model of the environment but learn directly from interactions with the environment. Recurrent PPO has recently enabled agents to use memory in partially observable environments [Pleines et al., 2022].

3. Proposal

This section outlines the proposed architecture for decision-making in a flexible multi-agent manufacturing environment. It introduces unit-specific service criteria, new communication mechanisms, updated demand statuses, and strategies for handling rejected demands through inter-agent transfers.

3.1. Intelligent Multi-Agent Manufacturing Network

The flowchart in Figure 1 shows a **self-optimised production system** where agents spread throughout the network prioritize and route demands through three specialized manufacturing units.

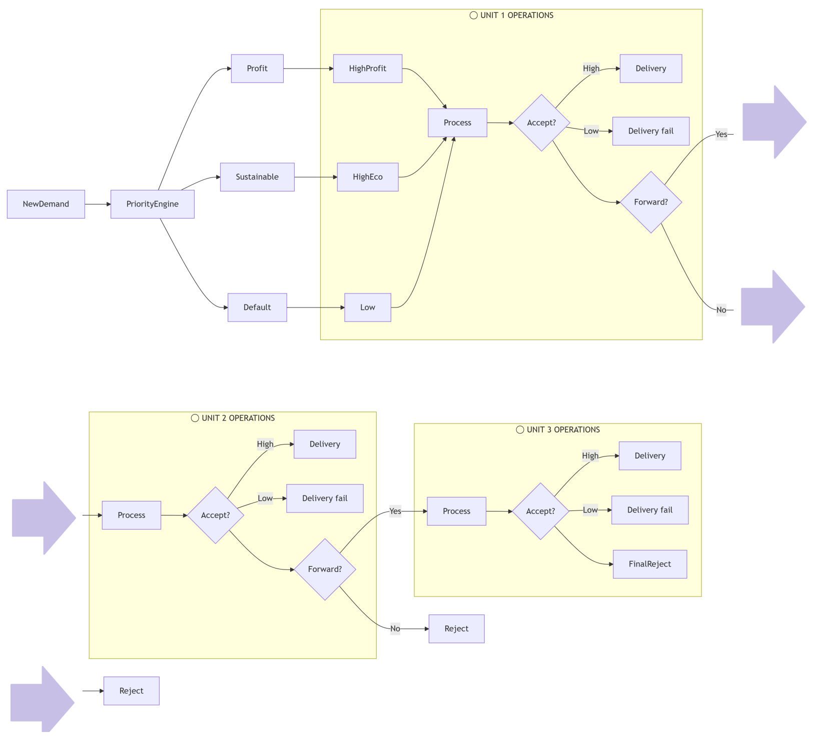

Figure 1: Flowchart of a 3-unit self-optimised production system, including the priority engine, decision-making agent, production, and delivery processes.

The three priority criteria, profit, sustainability, and variability, are used to classify the arriving demands in **Profit-Driven**, **Eco-Certified**, and **Custom-Made**. Each demand is then driven to a specific queue from which it is taken to start processing. The decision-making agents manage the dynamic routing logic, prioritizing the demand according to the unit profile or global three-criteria balancing. Agents evaluate both queue states and demand attributes, and high-score orders bypass lower ones unless the risk of delivery failure is exceeded. The whole system is self-optimized because all the flows are decided dynamically depending on the system performance under uncertainty about raw material availability, plant workload, and delays. A three-tier fallback system establishes cascading Fail-Safes, and final rejections occur only when the expected lead time is greater than the due time in all queues or there is no time remaining for forwarding to another unit.

Sustainable outcomes result from eco-prioritized orders and energy and resource savings due to rejection of those demands with a high risk of delivery failure. More traditional systems use storage yards extensively to keep products that have not been delivered. Besides, custom jobs route to specialized equipment first, and profit-critical demands maintain the Service Level Agreement (SLA) compliance.

The system **dynamically balances** economic, environmental, and customer-experience KPIs through distributed decision-making, with each agent applying local rules that collectively optimize global performance. Rejection patterns feed back into demand forecasting models for continuous improvement.

3.2. Procedure and Decision Rules

The manufacturing unit's local process can be isolated from the rest of the production network by considering specific interactions with local customers and suppliers. Thus, the decision-making process can be considerably simplified into a sequence of decisions to proceed with or interrupt the production stages of manufactured goods, depending on the risk of profit or loss under prevailing circumstances and uncertainties.

The decision policy for the priority queue is represented by hyperparameters associated with different classification criteria that can be applied depending on operational circumstances. Various rankings allow the system to be evaluated and used to achieve different objectives. Below, we define indices that offer different perspectives on the variables observed in the environment (Equations 5, 6, and 7). The values are normalized between 0 and 1, where values closer to 0 indicate higher priority for a given demand in the unit where it is being processed.

$$h_{1,d} = \frac{1}{AM_d \cdot PR} \tag{5}$$

$$h_{2,d} = 1 - VA_d \tag{6}$$

$$h_{3,d} = 1 - SU_d \tag{7}$$

The index h_1 is calculated for each demand d, considering the cost-to-profit ratio. Note that profit is net, as all production costs have been deducted. Index h_2 is an inverse proportion related to the variability of features (resource matrix) of a given order (or demand), where variability is a subjective concept calculated by summing the included feature vector (the more features, the greater the variability). Index h_3 applies the same logic to the estimated sustainability of a specific demand. The sustainability function is flexible enough to include aspects such as sustainable suppliers or carbon footprint, among other factors.

Figure 2 shows the state sequence for each received demand. Circles illustrate procedures and corresponding state values, while arrows connect states based on specific conditions. The lines drawn connect the final states, where a reward or penalty may be applied to the resulting demand.

The *Order&Receive* and *Released* procedures are part of the negotiation process with suppliers to obtain the raw materials needed to meet demand. These demands may be prioritized based on profitability and available resources, considering production risks such as supplier availability, machine reliability, and potential contractual penalties. Upon receiving a new demand, the *Order&Receive* procedure checks if it is already present in the yard by comparing it with others using a ”mask” that identifies the machines it has passed through. If a match is found with another existing demand, it is considered fulfilled and moves directly to the delivered state without going through planning and production.

When a demand is produced beyond the stipulated time, it is given the stored (*Store*) status and transferred to the yard. If the yard capacity is complete, the demand must be discarded, the most severe penalty, calculated according to Equation 12. The *Reject Local* procedure refers to the rejection of the demand by the unit and checks whether all other units have already rejected it. If so, the demand is rejected. Otherwise, a new destination is determined using the function defined in Equation 8 and illustrated in Figure 3.

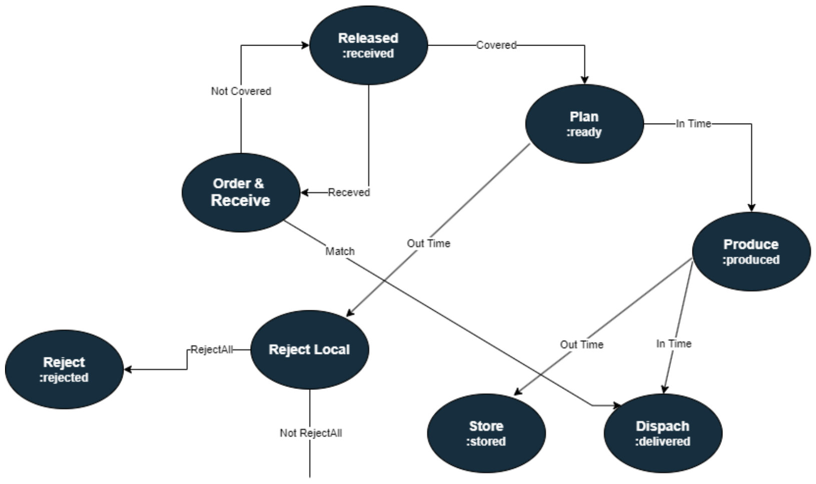

Figure 2: State space, including 4 operational states: received (Order-Receive-Released), production planning (Plan), locally rejected (Reject Local), produced (Produce), and 3 final states: rejected (Reject), stored (Store), and delivered (Dispatch)

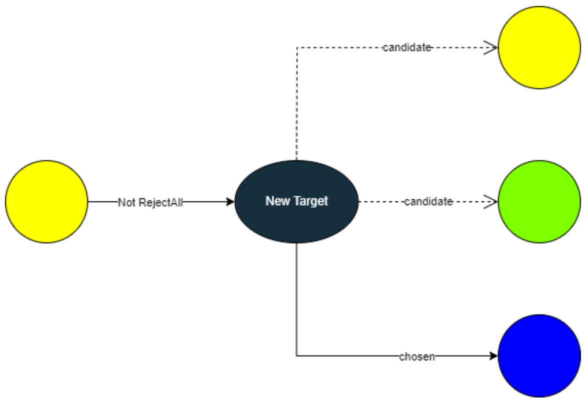

Figure 3: Each circle represents a peripheral unit in the system, and each color denotes its priority: yellow for profit, green for sustainability, and blue for variability

The *New Destination* procedure, illustrated in Figure 3, involves selecting a new peripheral unit to which a rejected demand is reallocated. This redirection of demand d uses Equations 5, 6, and 7. Based on these values, the system determines which priority queue best matches the demand, whether it aligns more with profit, sustainability, or variability, and reallocates it accordingly.

$$\text{New Target}(A, D) = \begin{cases} \arg\min_{i \in A}(\text{Queue}_i) & \text{if } P_d = \max(h_{1,\text{d}}, h_{2,\text{d}}, h_{3,\text{d}}) \text{ and } \text{ST}_d = -1 \\ -1 & \text{if no unit meets the conditions} \end{cases} \tag{8}$$

where A represents the manufacturing units in the system, P_d the normalized value of a demand's attributes, and ST_d its status.

In this study, risk was evaluated solely based on time constraints. The decision to proceed with production depends on whether the demand is within its delivery deadline. If the delay risk is acceptable, the demand is scheduled for production. The final states are Dispatch when delivery is successful, and Store when it is not.

3.3. Multicriteria Evaluation

There are several ways to design the objective function using a multicriteria approach. In a global context, it is desirable to balance the behavior of the entire manufacturing network by steering each decision toward favoring profit, variability, or sustainability criteria. In a local context, the goal may be more specific to fulfill regional requirements, such as legal regulations, local customer service, and eco-friendly suppliers. It is important to note that these objectives may conflict, as the manufacturing unit will likely incur additional costs to align its actions with local communities.

Therefore, we designed a prototype for the multicriteria objective function initially considering: profit, sustainability, and variability, all defined within the interval [0, 1]. The binary decision variable x_d represents the demand that is accepted, scheduled and delivered to the customer hub. The objective function is a normalized composition of three objectives:

$$\max \sum_{d \in \mathbb{N}} (PR_d \cdot x_d \quad + SU_d \cdot x_d \quad + VA_d \cdot x_d) \tag{9}$$

Concerning the flow shop constraints, the objective is subject to:

$$C_{\mathbf{d}_{\leq d}} + LT_d \cdot x_d \leq DO_d \cdot x_d \quad \text{for} \quad d \in \mathbb{N} \tag{10}$$

where $C_{\mathbf{d}_{\leq d}}$ is the production time for all demands **d** before d and converges to the global *makespan*, $C(max)$, after the last demand produced, excluding those that violate the time constraint represented by Equation 10.

The reward function is associated with a specific demand d reaching one of the success states: stored or delivered. The stored recovery refers to a demand that was successfully produced but missed the delivery deadline. The product is kept in stock and may still be used for a future order if matched. Note that storage yards are restricted, as in networked manufacturing models, energy waste and plants configured with large storage areas should be avoided.

One of the main challenges in customized production systems is reconciling specific demands with available stock, given the limited storage space. Accepting urgent orders can increase risks and should be avoided. Excess stock should only be used as a last resort and not encouraged as a standard practice. When space is available, storage should not generate a reward; a penalty should apply when it is not. Meanwhile, the delivery of customized products should be rewarded based on the nominal profit of the batch.

The reward RW of a specific unit is calculated according to its priority, defined by Equation 11, where penalties from poor decisions are not to be overlooked, as described in Equation 12.

$$RW_d = \begin{cases} RW_{pr} = AM \cdot PR & \text{if the unit prioritizes profit} \\ RW_{va} = AM \cdot PR \cdot VA & \text{if the unit prioritizes variability} \\ RW_{su} = AM \cdot PR \cdot SU & \text{if the unit prioritizes sustainability} \end{cases} \tag{11}$$

The penalty is considered when a demand is sent to production but not delivered on time. A demand that is delayed and stored in the yard is penalized geometrically by the current storage consumption SP_d of the yard $|YARD|$ (reaching zero in a nearly full yard). Equation 12 also considers the environmental tax **ET**, representing a penalty for yard use or waste status demands, which are regarded as violations of sustainability.

$$PE_d = \begin{cases} \frac{RW_{d\leq d}}{|YARD| - SP \cdot x_{d+1}} \cdot ET & \text{if demand } d \text{ has status :stored} \\ -RW \cdot ET & \text{if demand } d \text{ has waste status} \\ 0 & \text{if demand } d \text{ has any other status} \end{cases} \tag{12}$$

Equation 11 models the reward for a given episode. For a queue of demands processed before the current d ($d \leq d$), the updated reward for $d+1$ may be added by a factor proportional to the nominal profit $PR_d \cdot AM_d$, in the case of delivery, or be penalized according to Equation 12.

$$RW_{d+1} = \begin{cases} RW_d & \text{if demand } d \text{ has status :rejected} \\ RW_{d\leq d} - PE_{d\leq d} & \text{if demand } d \text{ has any other status} \end{cases} \tag{13}$$

The rejection of a production order is treated in this work as an operational decision based on the risk of failed delivery and is therefore not subject to penalties. The order is redirected to another unit in the manufacturing network scenario, following the decision policy described in Equation 8.

4. Computational Results

The experiments aim to validate a *framework* that integrates a multi-agent reinforcement learning (RL) interface with a manufacturing unit simulator. This simulator interacts with external sources to model suppliers and consumers, receiving orders with raw material specifications and deadlines. Each manufacturing unit decides whether it accepts a demand in the lineup and, if not, the demand is forwarded to another unit more suitable to accept it, considering the multi-criteria objective. The framework was implemented[1] with Python, leveraging RLlib 2.9.2 [Liang et al., 2018], PettingZoo 1.24.2 [Williams et al., 2021], Wandb 0.16.1 [Biewald, 2020], and Ray Tune 2.9.2 [Liaw et al., 2018].

Each test consists of 100 episodes (each with 4,000 interactions), with agents responsible for peripheral units. The number of agents varied from 1 to 6, with a previously established objective criterion evenly distributed as profit, customization, and sustainability. This strategy allows for the analysis of deadline and priority issues in the redirection of demands, highlighting that, in a real scenario, these priorities would be dynamic and balance economic and non-economic objectives.

As shown in Figure 4, rewards increase as more agents collaborate. Despite the evolution of environment rewards differing noticeably, it is not fair to evaluate only total rewards, considering the reward aggregation effect. Thus, we analyze agent 0, the only one present in all configurations, to provide a more consistent benchmark. Figure 5 reveals that increasing agent count reduces total rejected demands, confirming effective cooperation in demand redirection.

Figure 6 shows that more agents improve demand absorption, especially when aligned with demand features (e.g., profitability or sustainability). In the 6-agent scenario—where each objective is double-covered—transfers are more likely to succeed. As Figure 7 shows, most accepted transfers result in timely production.

Occasionally, transferred demands miss deadlines and are stored in the yard (Figure 8). Posteriorly, an incoming demand matches the unit's previously inventoried demand (produced, not delivered, and stored), allowing it to be immediately sent to the client [Ferreira et al., 2023]. Figure 9 details the number of demands assigned to agent 0 (always prioritizing profit in the 3-agent and 6-agent systems). Figure 10 quantifies such matches, increasing by 135% in the 6-agent case and 235% in the 3-agent case. These gains reflect increased transfers and higher system flexibility.

[1]https://github.com/alyssoncrvg/OPTSIMFLEX

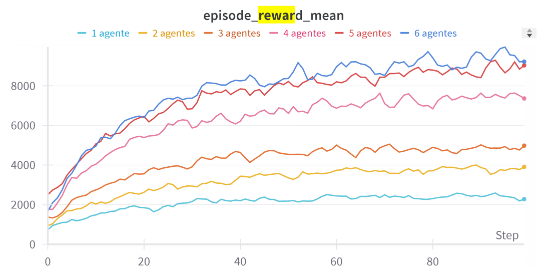

Figure 4: Total environment reward across scenarios with 1–6 agents

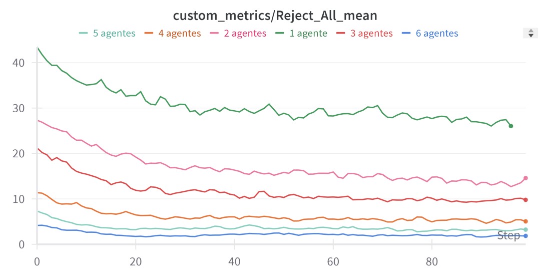

Figure 5: Total rejected demands vs. number of agents

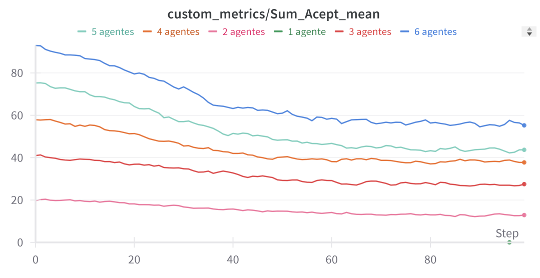

Figure 6: Accepted demands vs. number of agents

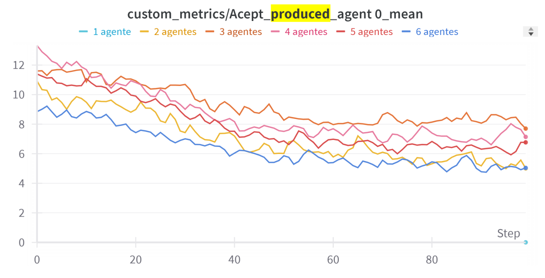

Figure 7: Transferred demands successfully produced

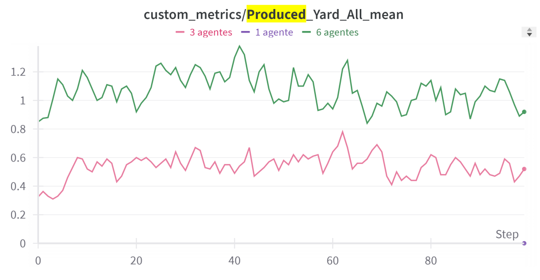

Figure 8: Transferred demands produced but delayed (stored in the yard)

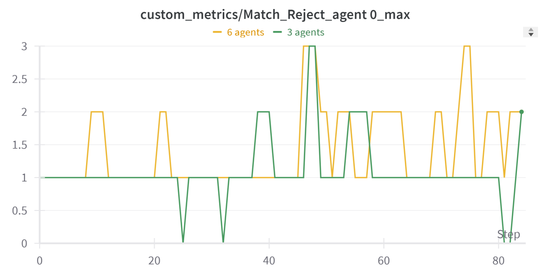

Figure 9: Transferred demands matched with yard stock (agent 0)

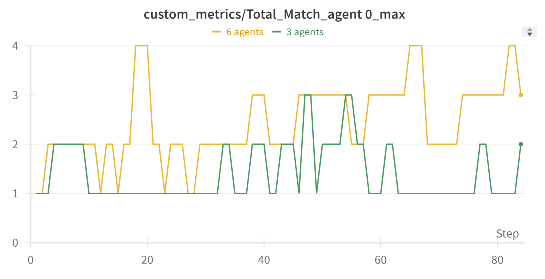

Figure 10: Total matches in agent 0's yard

All previous scenarios assume unlimited yard space. Without it, agents face stricter constraints and penalties. The following results compare such scenarios. Figure 11 shows that agents with access to yard space incur 15% fewer penalties on average. Figure 12 highlights waste as the main penalty source when no yard is available, with the 6-agent setup discarding over two demands per episode.

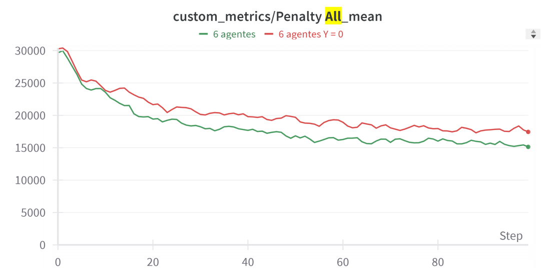

Figure 11: Average penalties with and without yard

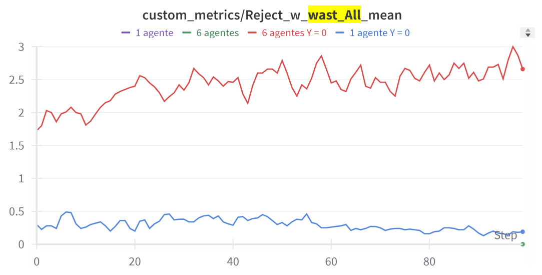

Figure 12: Average discarded demands

Even single-agent environments discard demands without yard access (Figure 13). Figure 14 shows a 20% increase in rejections when yard capacity is absent. This indicates that agents learn to avoid risk early during training.

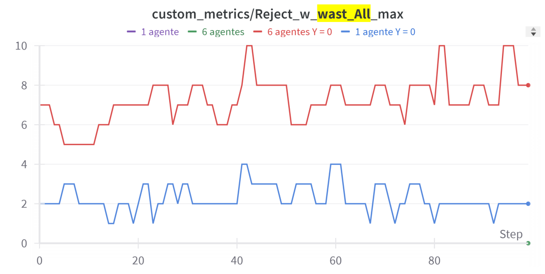

Figure 13: Maximum discarded demands per episode

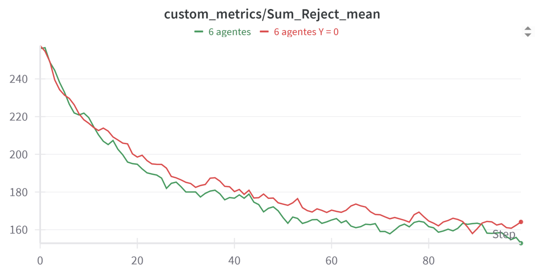

Figure 14: Rejected demands with and without yard

Despite increased rejections, the system maintains similar levels of accepted demands (Figure 15), demonstrating effective redirection to suitable agents. Nearly all accepted demands are ultimately produced and delivered (Figure 16), showing a refined balance between caution and efficiency without yard storage.

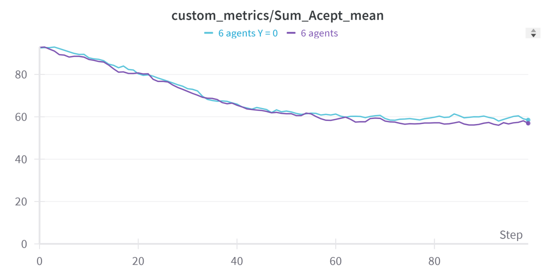

Figure 15: Accepted demands (no yard)

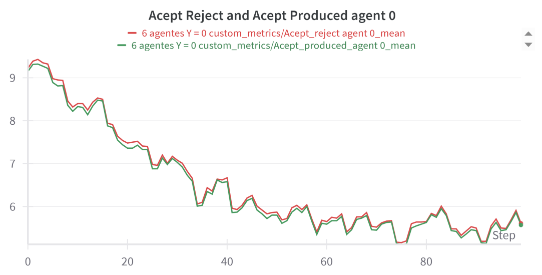

Figure 16: Demand handling breakdown (no yard)

5. Conclusion

This study presented a framework for integrating reinforcement learning agents in a multi-agent environment simulating manufacturing units, oriented toward personalized production. The main objective is to develop a structure capable of representing agents with multiple goals (profitability, sustainability, variability), reflecting the demands of Industry 4.0. The results demonstrate that the inclusion of multiple agents with distinct objectives increases the system's ability to handle a broader range of demands, fostering both collaboration and specialization. The proposed redistribution mechanism helps mitigate order rejections and leverages idle capacity in compatible units, enhancing the system's overall efficiency.

Furthermore, the study evaluated scenarios both with and without buffer storage (yard), highlighting its critical role in absorbing uncertainty and increasing operational flexibility. Even in the absence of a yard, agents were able to adapt their behavior through learning, minimizing penalties associated with product waste and adjusting their demand acceptance strategies accordingly. The tools and libraries employed—namely RLlib, PettingZoo, Wandb, and Ray Tune—played a key role in the training, evaluation, and tuning of the learning models, enabling robust and reproducible simulations.

For future work, the following directions are suggested: a) Introducing variability in demand arrival rates to simulate seasonality and external events; b) Incorporating explicit communication between agents to enable coordinated decision-making; c) Including organizational and social well-being metrics as agent objectives; d) Adapting the framework to real cyber-physical systems to assess its practical viability. The findings indicate that reinforcement learning-based multi-agent approaches are promising for the design of flexible, collaborative, and resilient manufacturing environments, aligned with the principles of Industry 4.0 and 5.0.

Acknowledgements This work was partially supported by Marie Skłodowska-Curie (847.577), Science Foundation Ireland (CONFIRM 16/RC/3918 and Insight 12/RC/2289-P2), and scholarship by FAPEMA (Proc. BIC-05014/24), Brazil.

References

Abualigah, L., Hanandeh, E. S., Zitar, R. A., Thanh, C.-L., Khatir, S., and Gandomi, A. H. (2023). Revolutionizing sustainable supply chain management: A review of metaheuristics. *Engineering Applications of Artificial Intelligence*, 126:106839. ISSN 0952-1976. URL `https://www.sciencedirect.com/science/article/pii/S0952197623010230`.

Biewald, L. (2020). Experiment tracking with weights and biases, 2020. *Software available from wandb. com*, 2(5).

Cheng, C., Ahmad, S. F., Irshad, M., Alsanie, G., Khan, Y., Ahmad, A. Y. B., and Aleemi, A. R. (2023). Impact of green process innovation and productivity on sustainability: The moderating role of environmental awareness. *Sustainability*, 15(17):12945.

Dennison, M. S., Kumar, M. B., and Jebabalan, S. K. (2024). Realization of circular economy principles in manufacturing: obstacles, advancements, and routes to achieve a sustainable industry transformation. *Discover Sustainability*, 5(1):438.

Ferreira, F. M., de Souza, B. F., de Oliveira, A. C. M., and Brown, K. N. (2023). Self-optimising manufacturing network: an online optimization problem decided by reinforcement learning. In *LV Simpósio Brasileiro de Pesquisa Operacional*.

Kim, Y. G., Lee, S., Son, J., Bae, H., and Chung, B. D. (2020). Multi-agent system and reinforcement learning approach for distributed intelligence in a flexible smart manufacturing system. *Journal of Manufacturing Systems*, 57:440–450.

Li, C., Zheng, P., Yin, Y., Wang, B., and Wang, L. (2023). Deep reinforcement learning in smart manufacturing: A review and prospects. *CIRP Journal of Manufacturing Science and Technology*, 40:75–101.

Liang, E., Liaw, R., Moritz, P., Nishihara, R., Fox, R., Goldberg, K., Gonzalez, J. E., Jordan, M. I., and Stoica, I. (2018). Rllib: Abstractions for distributed reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning (ICML 2018)*. URL `https://arxiv.org/abs/1712.09381`. arXiv:1712.09381.

Liaw, R., Liang, E., Nishihara, R., Moritz, P., Gonzalez, J. E., and Stoica, I. (2018). Tune: A research platform for distributed model selection and training. *arXiv preprint arXiv:1807.05118*.

Lu, Y., Zhang, Y., Li, Y., et al. (2021). Self-organizing manufacturing network: A comprehensive review. *Journal of Manufacturing Systems*, 60:35–47.

Pleines, M., Pallasch, M., Zimmer, F., and Preuss, M. (2022). Generalization, mayhems and limits in recurrent proximal policy optimization. *arXiv preprint arXiv:2205.11104*.

Popper, A., Silva, M., and Torres, R. (2023). A multi-criteria dynamic planning of production facilities based on ppo in a multi-agent rl setting. In *Proceedings of the International Conference on Smart Manufacturing*, p. 210–223. Springer.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017a). Proximal policy optimization algorithms.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017b). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Váncza, J., Monostori, L., Erdős, G., and Kádár, B. (2019). Adaptive production planning and scheduling in industry 4.0. In *IFIP International Conference on Advances in Production Management Systems*, p. 1–8. Springer.

Wang, Z., Zhang, H., and Liu, Y. (2021). Reinforcement learning for real-time production scheduling in smart factories. *Journal of Manufacturing Systems*, 59:1–13.

Williams, N., Lokesh, Y., and Ravi, P. (2021). Pettingzoo: Gym for multi-agent reinforcement learning. *Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS 2021)*. URL `https://www.pettingzoo.ml`. Available at: `https://github.com/Farama-Foundation/PettingZoo`.

Zhang, C. and Gao, L. (2020). Multi-objective optimization in flexible manufacturing systems using deep reinforcement learning. *Computers & Industrial Engineering*, 147:106664.

Zhang, Y., Zhu, H., Tang, D., Zhou, T., and Gui, Y. (2022). Dynamic job shop scheduling based on deep reinforcement learning for multi-agent manufacturing systems. *Robotics and Computer-Integrated Manufacturing*, 78.

Zhou, Q., Wu, Y., and Liu, X. (2021). A multi-agent deep reinforcement learning approach for dynamic resource allocation. *IEEE Access*, 9:8172–8184.