São José dos Campos - SP
LV SBPO
LV Simpósio Brasileiro de Pesquisa Operacional
São José dos Campos, SP - 6 a 9 de novembro de 2023

Self-Optimising Manufacturing Network: an Online Optimisation Problem Decided by Reinforcement Learning

Frederic Menezes Ferreira
Computer Science Graduate Programme, Federal University of Maranhão
Bancaga. São Luís, MA, Brazil
frederic.menezes@ufma.br

Bruno Feres de Souza
Computer Engineering Department, Federal University of Maranhão
Bancaga. São Luís, MA, Brazil
bruno.feres@ufma.br

Alexandre Cesar Muniz de Oliveira
Department of Informatics
Federal University of Maranhão
Bancaga. São Luís, MA, Brazil
alexandre.cesar@ufma.br
Confirm Centre for Smart Manufacturing
University College Cork
Western Gateway Building, Cork, Ireland
amunizdeoliveira@ucc.ie

Kenneth N. Brown
School of Computer Science and Information Technology
University College Cork
Western Gateway Building, Cork, Ireland
k.brown@cs.ucc.ie

ABSTRACT

Networked personalised production necessitates resilient optimisation approaches in the face of uncertainty to proactively react to unforeseen scenarios and enable early action to address sudden shifts in consumer and supplier connections. This work presents a comprehensive framework for decision-making within a local unit of a self-optimising manufacturing network, explicitly focusing on exploring the interplay between optimisation, simulation, and reinforcement learning. We propose an initial validation scenario in which the agent needs to accept production demands considering priorities concerning three performance criteria (economic, sustainability, and variability), given the current load on the system that can impact delays and, consequently, financial losses. Following the hyper-parameter setup experiment, the computational results strongly support validating the manufacturing unit simulator and the Proximal Policy Optimisation-based decision agent.

KEYWORDS. Self-optimising manufacturing network. Reinforcement learning. Recurrent Proximal Policy Optimisation.

DOI: 10.59254/sbpo-2023-175124
http://live-proceedings.pantheonsite.io/p/175124

1. Introduction

Mass customisation allows customers to select from various pre-defined product configurations. In contrast, mass personalisation works with individual customers to specify products tailored to their preferences [Tiihonen e Felfernig, 2017]. This changes the manufacturing paradigm from high volume and low variety to low volume and high sort, down to batch size one in ultra-personalised products and services (UPPS) [Nachtigall et al., 2020]. Penalisation allows businesses to compete on added value rather than solely on cost.

In addition, there is a broader trend towards sustainable production and logistics, requiring localised manufacturing, in which items are produced closer to the customer. Localised production brings further challenges, as the market conditions, the availability of raw materials, and the operation of the supply networks all may become more unstable, which further increases the need for dynamic and flexible production [Zheng et al., 2021].

A self-organising manufacturing network (SOMN) is a new organisational concept designed to allow personalised products with dynamic demand, relying on a flexible structure and adaptive control. The SOMN's structural characteristics are [Qin e Lu, 2021]: a) manufacturing units are connected in a peer-to-peer fashion (**Distributed**); b) control decisions are generated via local negotiations and coordination between manufacturing units, not accepting commands from the top (**Bottom-up**); c) production volume and product variety can be dynamic and adjusted in response to demand (**Adaptability**); d) manufacturing units decide on precise configurations and localised tasks based on peer-to-peer communication and negotiation (**Individual autonomy**).

Optimisation in a SOMN refers to providing control of the production processes from economic-related indicators (makespan, resource utilisation, workload, profit, and costs), sustainability (energy consumption, waste reduction, CO2 emissions), social engagement and system-level performances (adaptability and scalability) [Azarakhsh et al., 2021].

In this sense, Reinforcement Learning (RL) has presented promising results in solving sequential decision-making problems in complex environments and providing a highly adaptable solution [Cunha et al., 2018; Zhang et al., 2022; Chang et al., 2022; Wang et al., 2022]. Existing RL algorithms perform poorly due to the significant exploration required, which can be prohibitively costly and time-consuming for online learning. Consequently, RL is predominantly utilised for offline learning, simulating realistic production environments[Sutton e Barto, 2018].

Proximal Policy Optimisation (PPO) is presently considered state-of-the-art in policy gradient methods for RL, employing an objective function that enables multiple epochs of mini-batch updates rather than performing only one gradient update per data sample, also with good computational speed and robustness [Schulman et al., 2017]. Additionally, Recurrent PPO has recently enabled agents to use memory in partially observable environments [Pleines et al., 2022].

This paper introduces a framework for multi-criteria decision-making in SOMN, leveraging Reinforcement Learning (RL) techniques. The proposed framework addresses uncertainty while effectively balancing economic, local engagement, and sustainability indicators. We present a simulator for a manufacturing unit designed to be integrated into a broader network. The results showcase the Recurrent PPO-based RL agent's capability to learn effective control policies for managing production acceptance risk in the face of different performance criteria.

This paper is organised as follows: Section 2 motivates the reinforcement learning for problems related to SOMN. Section 3 presents the proposed framework, including the environment modelling and evaluation functions. In Section 4, the computational results are discussed, and Section 5 highlights the main findings and future research.

DOI: 10.59254/sbpo-2023-175124
http://live-proceedings.pantheonsite.io/p/175124

2. Reinforcement learning role

Reinforcement Learning (RL) is an algorithmic approach that leverages trial-and-error to find an optimal decision-making policy [Kim et al., 2020]. This policy serves as a function mapping from the state to the action space, guiding the RL agent to select appropriate actions based on the perceived state of the environment [Sutton e Barto, 2018]. In the context of a manufacturing system, the environment can be simulated as a predefined state machine that evolves based on stochastic input data. The RL agent learns and refines its decision-making map by anticipating rewards through the feedback received that is used to evaluate the performance of the agent's actions and optimise the policy iteratively until convergence is achieved [Zhang et al., 2022].

Deep Reinforcement Learning has emerged as a valuable approach in various domains, aiming to achieve fast response and accuracy by learning decision-making processes while enabling reusability and expandability based on past experiences. Deep agents have demonstrated superior performance to traditional dispatch rules such as first-in-first-out, profit maximisation, and penalty minimisation, rapidly converging to above-average levels. Notably, the *Duelling Double Q-Network* has exhibited advanced stability and optimal behaviour, irrespective of increasing training episodes [Tang e Salonitis, 2021].

In a multi-objective setting, the reward function can employ a weighted composition, where different objectives regulate the schedule's performance and facilitate knowledge acquisition to enhance their experience. For instance, in [Zhou et al., 2022], composite rewards account for time-saving, energy-saving, machine utilisation, and workload distribution deviation. However, only some studies have explored adaptive control approaches for manufacturing systems that encompass all criteria without relying on weighted rewards.

Our approach distinguishes itself from previous works by considering the influence of decisions on future choices and relationships with suppliers and customers. Nevertheless, the core decision-making on the shop floor can be effectively represented using the model proposed in [Zhou et al., 2022]

3. Framework proposal

Considering the literature review about Personalised-Mass Production and Self-Organised Manufacturing Networks, we are focusing on enabling technology based on RL to provide an initial framework for making production scheduling decisions.

3.1. Environment modelling

The network of manufacturing units is a complex system with at least two specific functions: peripheral and backbone units. For simplicity, we assume a backbone unit as a central manufacturer that applies structural features of interest to many worldwide customers (mass-production stage). Locating backbone units near big distribution hubs with facilities to reach other units in the network is strategic. Conversely, a peripheral unit is responsible for printing specific and regional features (personalised stage). Locating peripheral units near customer markets and regional supply chains is also strategic. Indeed, we are dealing with flows of semi-finished products, a set of local suppliers, and distribution hubs, pursuing local objectives (rewards) and balancing overall multi-criteria objectives (control).

Specific assumptions We are assuming an environment composed of autonomous manufacturing units negotiating raw material amounts and deadlines to face with foreseen demands of personalised products. Each manufacturing unit contributes with its production activity to the global multi-criteria objective regarding costs, profit, and indicators related to sustainability issues. For simplicity, no predatory competition among the manufacturing units is permitted. We understand

DOI: 10.59254/sbpo-2023-175124
http://live-proceedings.pantheonsite.io/p/175124

this side effect can be hypothetically prevented by not allowing price negotiation and raw material containment to avoid units choosing which demand to supply.

Concerns about variability are introduced by specific and regional features aggregated to structural ones. So, we can design a final product as a sequence of manufacturing units where it passes. In Figure 1, the manufacturing network's overall context is depicted as a set of units topologically connected. Each coloured line means a sequence of units responsible for aggregating a specific feature set and introducing product variety (personalisation). All lines start from a backbone manufacturing unit (orange square) and finish at a customer hub (green triangle). For example, the products *P41* and *P42* can be assembled (or represented) by the following sequences, $\{A2, B, D1, E\}$ and $\{A3, C, D2, F\}$ respectively.

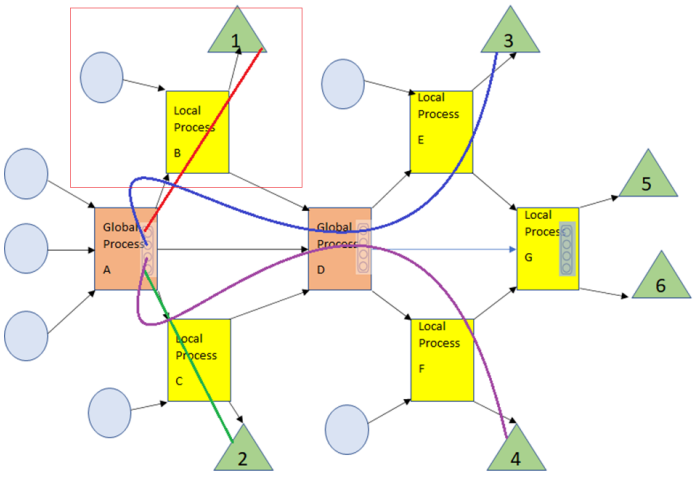

Figura 1: Manufacturing Network: two units dedicated only to structural features (backbone), five units dealing directly with customer hubs (triangles) and suppliers (circles). Production flows through the manufacturing network showing sequences 2-assembled products $\{A1, B\}$, $\{A4, C\}$ and 4-assembled ones $\{A2, B, D1, E\}$, $\{A3, C, D2, F\}$. Clipping of the whole manufacturing network representing a peripheral unit and the local production flow

The placement of units is not the objective of this work but only to decide supply chain flows and product delivery. We are focusing on a specific peripheral unit (Unit B, for example) since it has to deal with regional suppliers and customers, i.e., its ecosystem is more heterogeneous, as shown in Figure 1.

Sequential Decision Process In sequential decision problems, the utility function is obtained from the state reached by the agent's actions. Each action is made from a set of available decisions and is based on a decision rule under certain circumstances, considering complete information of the system state. An immediate utility value is obtained for each decision, and the system transits to a new state. The objective of a sequential decision problem is to find the optimal policy, i.e., a sequence of decision rules for each future time step [Strøm et al., 2018].

We are dealing with a simplified inventory considering the balance, `BA[1..M]`, for M *glocal* features (global and local) inventoried at the local unit [Rauch et al., 2017]. The *balance* is used in the decision process since we can prioritise product demands for which the balance is sufficient to cover the production with minimum need of ordering raw material. The whole inventory regards not only in-site raw materials but such ones that are incoming, `IN[1..M]`, already ordered,

DOI: 10.59254/sbpo-2023-175124
http://live-proceedings.pantheonsite.io/p/175124

but not received) as well as those already expected to be employed in the production lineup, i.e., `OU[1..M]`. The availability `AV[1..M]` of the whole inventory can be calculated by:

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

The decision to order a missing raw material is taken over the availability AV if the matching function fails to find a similar product in the yard ready to be delivered. *Demands* are received at specific time windows in batches from each customer hub identified by an attribute CU. Time stamping is used to sort the demands in the system, considering the input time, DI, and output due time, DO. Recalling DO must cover the product's expected lead time LT. In this work, the demands are organised in a First-In-First-Out (FIFO) scheme.

Each demand batch has its expected cost CO, space consumption SP if kept in the yard, net profit PR for each unit requested AM, and some penalty PE for accepting and not delivering the batch. ST and $FT[1..M]$ refer to the demand status and the feature array, respectively. The feature array works as a personalised signature defining the variability degree of the demand. In job-shop problems, FT represents the machine time for each feature to be printed in the product. The cost CO_d and the space factor are calculated considering the amount AM_d as shown in 2.

$$CO_d = AM_d \cdot FT_d \cdot EU \tag{2}$$

where EU is the expected price of each raw material included in the FT_d. The remaining demand attributes LT,VA, and SU are computed by specific functions shown in equations 3 to 5.

$$LT_d = \tau(AM_d \cdot FT_d) \tag{3}$$

where τ is a function regarding the amount, features, and current time expectations for production. Variability can be used to demonstrate a good engagement with a diverse public since small feature changes can make a product most sought after by a group of people. Although a given demand is costly due to the high lead time or raw material prices, it is possible to reach a good variability.

$$VA_d = \upsilon(FT_d) \tag{4}$$

Sustainability is calculated regarding the carbon print, energy consumption, environmental impact, or something equivalent. A general rule is that the more features are aggregated into the finished product, the more environmentally impactful and less sustainability is achieved.

$$SU_d = \sigma(FT_d^{-1}) \tag{5}$$

A demand batch can assume different statuses during the production process, varying from received(0), ready(1), rejected(2), produced(3), stored(4), and delivered(5).

$$ST_d = \{0, 1, 2, 3, 4, 5\} \tag{6}$$

Procedures and decision rules The local process under investigation in this work can be isolated from the remaining manufacturing network, considering a few interactions with local customers and suppliers. Therefore, the decision-making process can be significantly simplified to a decision sequence that advances or cancels the production stages of manufactured goods depending on the risk of profit or loss in the face of circumstances and uncertainties.

DOI: 10.59254/sbpo-2023-175124
http://live-proceedings.pantheonsite.io/p/175124

In Figure 2, one can see the sequence of states for each demand received. Circles represent procedures and state values assumed, while arrows connect states depending on certain conditions. Traced lines connect final states in which a reward or penalty can be applied to demands reaching.

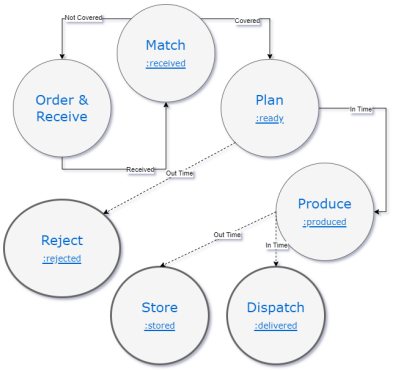

Figura 2: State space, including 4 operational states: received(Order-Receive-Match), ready(Plan), produced(Produce); and 3 final states: rejected(Reject), stored(Store), and delivered(Dispatch)

Procedures *Order&Receive* and *Match* compose the negotiation procedure, in which suppliers are contacted to provide raw materials needed to cover the demanded products. All demands arriving can be managed to be prioritised by those more promising, considering profit and resources available to meet production. Decision-making revolves around the risk inherent in the production process that depends on many factors, such as the availability of suppliers, reliability of processes and machines that take up space in the plant itself, as well as penalties and fines formatted in contracts [Heckmann et al., 2015; Silbermayr e Minner, 2014].

In this work, we used the definition of risk, considering the time factor exclusively. As Figure 2 shows, the decision to proceed with the production schedule depends solely on the time factor (`in Time` or `out Time`). Since a given demand reaches the *ready state* (*Plan state*) and the risk of out-time production is acceptable, the order is put in the production lineup (scheduled). *Dispatch* and *Store* are final states for a finalised production but differentiable by well-succeeded delivery or not, respectively.

Priority queue Decision policy has been represented by hyper-parameters associated with different decision-making ranking criteria that can be used depending on operational circumstances. The various rankings induce the evaluation of the system and can be used to pursue other goals. In the following, indexes are defined to offer different views over the environment's observed variables (Equations 7 to 10). The lesser the value computed, the greater the priority for a given demand.

$$h_{1,d} = \frac{CO_d}{PR_d} \tag{7}$$

$$h_{2,d} = \frac{1}{VA_d} \tag{8}$$

$$h_{3,d} = \frac{1}{SU_d} \tag{9}$$

$$h_{4,d} = \frac{1}{(DO_d - LT_d - t + 1)} \tag{10}$$

DOI: 10.59254/sbpo-2023-175124
http://live-proceedings.pantheonsite.io/p/175124

The index h_1 is computed for all demand d considering the rate between cost and profit. Recalling that profit is already discounted from all production costs, i.e., it is a net profit. The index h_2 is an inverse proportion for the feature variety of a given order. Recalling that variety is a subjective concept calculated by summing up the included feature vector (the more features, the more variety included). The index h_3 makes the same for the presumed sustainability of a given demand. Recalling that variety is a subjective concept calculated by a sustainability function, flexible enough to incorporate issues related to green-agenda suppliers or carbon footprint, as examples. The index h_4 considers the slack time to formulate the delivery failure risk as an inverse proportion of the remaining time, from the instant t, to produce a given demand, considering the due time DO_d, and discounted the expected lead time LT_d.

3.2. Multi-criteria evaluation

There are many possibilities to design the objective function in a multi-criteria way. In the global context, balancing the behaviour of the entire manufacturing network is desirable, leaning each decision step to favour the objectives of profit or sustainability. In the local context, the purpose goes further to deal with requirements associated with, for example, law regulations, local customer attendance, and green suppliers. Of course, these are antagonistic goals as the manufacturing unit is likely to pay more for each action aimed at harmonising with local communities.

Objective function and constraints Hence, we design a prototype for the multi-criteria objective function considering initially profit, sustainability, and local customer featuring attendance, all defined in the interval $[0, 1]$. As defined, the evaluation function depends on qualifying demand batches produced and delivered. We can represent the decision variable as:

$$x_d = \begin{cases} _{:produced} & \text{if} \quad \text{demand is to be produced} \\ _{:delivered} & \text{if} \quad \text{demand is to be produced and delivered} \\ 0 & otherwise \end{cases} \tag{11}$$

For simplicity, we define the objective as an unweighted result of 3 different criteria: the profit for each demand batch with delivered status; sustainability computed over every demand batch already scheduled to the lineup (delivered and stored); and customer featuring attendance, associated with the variability degree. Since the featuring array is designed to meet local customer requirements, the more features aggregated, the more specific the customer reached. We define the objective function as a non-normalised composition of three objectives:

$$\max \sum_{x_d \in ST =_{:delivered}} PR \cdot x_d \quad + \sum_{x_d \in ST =_{:produced}} (SU \cdot x_d + VA \cdot x_d) \tag{12}$$

Not excluding other typical flow shop constraints, the objective function is subject to

$$C(t)_{\mathbf{d} \leq d} + LT \cdot x_d \leq \max(0, DO \cdot x_d - t) \quad \text{for} \quad x_d \in ST =_{:produced} \tag{13}$$

where $C(t)_{\mathbf{d} \leq d}$ is the effective producing time for all demands $\mathbf{d}$ before d and converges to the global makespan, $C(max)$, after the last produced demand, excluding those that violate the time constraint represented by Eq.13.

DOI: 10.59254/sbpo-2023-175124
http://live-proceedings.pantheonsite.io/p/175124

Reward The reward function is associated with a given demand d reaching one of the successful states: stored or delivered. Recalling stored is a status related to well-succeeded demand production, which lost the deadline to be dispatched. The product is so updated in the inventory that it can be delivered in a future order.

Regarding customised production systems, one key challenge is matching specific incoming demands with existing inventory. Simply stocking up on batches may not be beneficial without limited storage space. Instead, it is important to avoid accepting rush orders that can lead to high-stranding risks. If storage space is available, excess inventory could be a last-resort option. However, this should not be encouraged as a policy, and the decision should be based on careful evaluation of risks and benefits. When there is storage space, storing the excess inventory should only be rewarded. If no storage space is available, it should be penalised proportionally for violating the space constraint. On the other hand, the reward for delivering customised products must be based on the nominal profit defined for the lot.

Equation 14 models the economic reward only, considering the three feasible states: stored, delivered, and rejected, in a given episode. Regarding a lineup of demands processed before current d ($\mathbf{d} \leq d$), the updated reward **d+1** can be added by a factor proportional to the nominal profit $PR_d \cdot AM_d$, in case of delivery (*:delivered* state), or be penalised geometrically by the current consumption SP_d of the yard space $|YARD|$ (reaching zero for an almost overflown yard situation), in case of storing (*:stored* state).

$$RW_{\mathbf{d+1}} = \begin{cases} \frac{RW_{\mathbf{d} \leq d}}{|YARD| - SP \cdot x_d + 1} & \text{if demand d is } :stored \\ RW_{\mathbf{d} \leq d} + PR \cdot x_d \cdot AM \cdot x_d & \text{if demand d is } :delivered \\ RW_{\mathbf{d}} & \text{if demand d is } :rejected \end{cases} \tag{14}$$

Production order rejection is treated in this work as an operational decision based on the risk of delivery failure and, therefore, not subject to penalty. In a manufacturing network scenario, there may be alternatives, such as forwarding the demand to another unit for which raw material is available. In an additive manufacturing scenario, transportation times could be neglected.

4. Computational results

The experiments are intended to validate the framework, including the RL interface and the manufacturing unit simulator that connects to two external sources: local providers and consumers. The unit receives demand orders with raw materials and delivery time requirements in each timestamp. It must accept to produce and deliver orders based on its raw material stocks, including incoming and outgoing forecasts, yard occupancy, and expected profit.

4.1. Code and Data

The framework is implemented in *Python* language (version 3.10.11), employing stable-baselines3 library [Raffin et al., 2021], OpenAI Gym toolkit [Brockman et al., 2016], for the simulator (environment) and the *Weights & Biases* (WandB [Biewald, 2020]) package to perform the hyper-parameter search and log all results. The agent learns an optimal policy using a learning algorithm, the Proximal Policy Optimisation (PPO) [Schulman et al., 2017].

The SOMN simulator is flexible in incorporating different system profiles by adjusting parameters. The main parameters are the storage capacity(`YA`), the number of features or raw materials (`M`), the demand set size (`N`), and the upper limits for the following parameters: Due time

DOI: 10.59254/sbpo-2023-175124
http://live-proceedings.pantheonsite.io/p/175124

(MAXDO), batch size (MAXAM), profit factor (MAXPR), rejection penalty (MAXPE), feature machine time (MAXFT), feature cost (MAXEU). We elect YA, M, and N related to the instance size. In this work, we set the instance with {YA=10, M=5, N=7,MAXDO=10, MAXAM=5, MAXPR=1.5, MAXPE=10, MAXFT=5, MAXEU=10}. Several instances are randomly generated over these parameters at each experiment depending on the number of episodes.

4.2. Hyper-parameters setup

The first set of experiments automates the hyper-parameter search and explores the space of possible learning models. We also compare two agents (Proximal Policy Optimisation - PPO and Recurrent PPO) in designing the reward function, using the best parameter set from the previous experiment. To find the best PPO and Recurrent PPO parameters, 82 train experiments identified the parameters associated with the highest reward and lowest loss for both agents, using WandB. The best hyper-parameters are presented in Table 1.

Tabela 1: Best configuration to train and test PPO and Recurrent PPO agents

Parameter	Recurrent PPO	PPO
gamma	0.8332421508315541	0.918623650457
n_steps	2048	3328
ent_coef	0.000819163636857	0.001641577520
n_epochs	17	21
target_kl	0.028836888292401	0.021139104464
batch_size	256	256
gae_lambda	0.953618470281289	0.914295046604
learning_rate	0.000112665315689	0.000366014479

4.3. Comparison between PPO and Recurrent PPO

To compare the PPO and Recurrent PPO hyper-parameterised agents, we performed an additional 10 training sessions. The sessions lasted 1,000,000 time steps for the two previously parameterised agents. Figure 3 depicts the performance obtained regarding reward and loss functions over time. As expected, the Recurrent PPO agent was slightly superior in both functions.

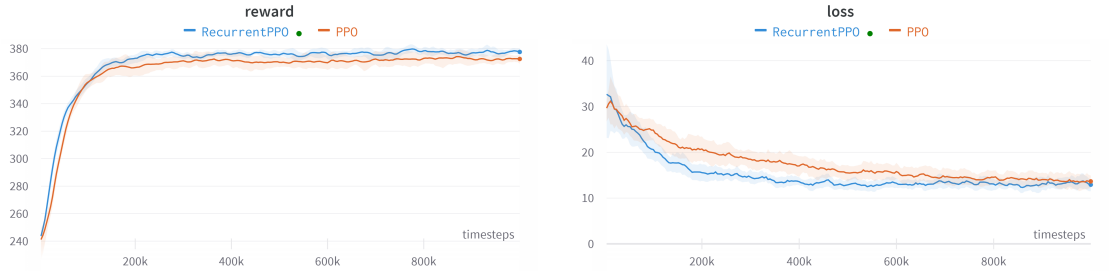

Figura 3: PPO and Recurrent PPO performances considering reward [360,380] and loss [10,20] over 1,000k time steps

4.4. Action space and Control

The acceptance risk is inversely proportional to the slack time (10) and is unknown since it depends on the expected lead time LT, including probabilistic delays. The probabilistic delay is modelled by Poisson's distribution, centred in $LT_d + \mathcal{L}(t)$, where $\mathcal{L}(t)$ is the number of active demands on production in time t. A valid result is an agent predicting/learning the delay and

DOI: 10.59254/sbpo-2023-175124
http://live-proceedings.pantheonsite.io/p/175124

disclosing the probability function. If the agent overestimates the delay, it will likely reject requests that could be completed in time to generate profit and reward. Otherwise, underestimating, the agent accepts high-risk demands that may not be delivered and will take up valuable space in the yard.

Figure 4 shows the performance of the PPO agent for different delay profiles. It is observed that the demand acceptance actions represent how the agent understands the delay probability distribution. In this case, the delays were particularly fixed (1,3,5,7,9,15, respectively) to allow the observation of the actions responding coherently. It is noticeable that the agent tries to follow the delay but always with risks underestimation, supposedly seeking to increase the reward for profit. Besides, for higher delay profiles, the system tends to lose performance regarding the reward obtained in each time step.

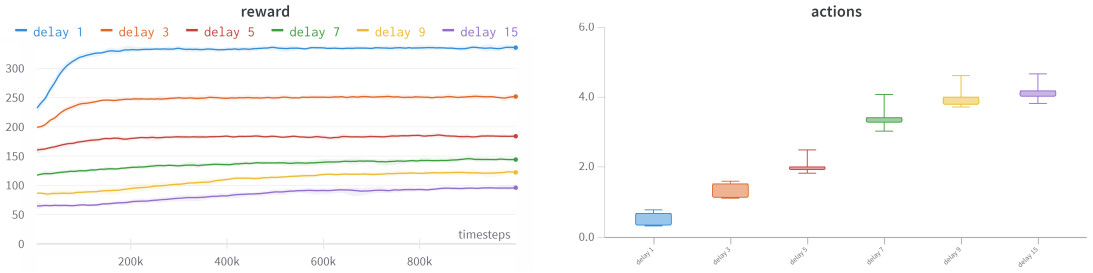

Figura 4: PPO agent reacting to fixed delay profiles 1,3,5,7,9, and 15: actions underestimate the delay, and the reward function decays to $1/6$ when delay rises from 1 to 15.

We also evaluated the ability to control performance criteria by prioritising demands with different characteristics. Profit and variability are interrelated since highly featured products are generally more expensive and must have higher built-in yields. Rather, sustainability requires more straightforward products, not requiring many production hours, and consuming fewer resources and energy. Higher profit rates are expected by prioritising the most profitable demands.

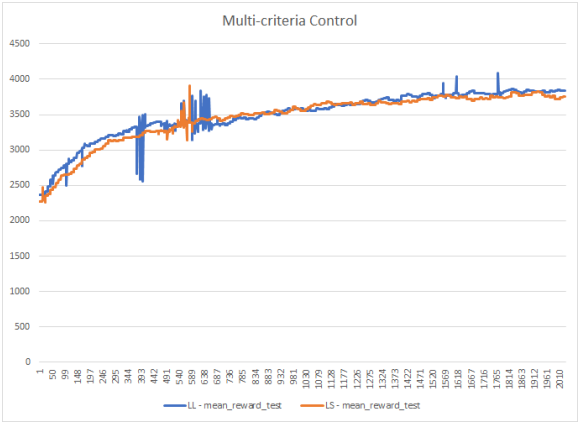

Figura 5: Average reward obtained in 20 executions, prioritising profit (LL) and sustainability (LS) demands: LL reached a profit of 78.83% better, totalling 0.72% of better decisions.

Figure 5 depicts the average reward obtained in 20 executions, with 10 prioritising profit (LL) and 10 sustainability (LS). We can observe that the two measures are equivalent during some time intervals, but the reward obtained by LL is higher on average, reaching $3,501.52 \pm 252.98$

DOI: 10.59254/sbpo-2023-175124
http://live-proceedings.pantheonsite.io/p/175124

against $3,461.68 \pm 266.24$ (a result of $80,863.36$, i.e., 3.72 better). The manufacturing unit simulator and the decision-making RL agents based on the PPO and Recurrent PPO presented results compatible with what is expected in this first battery of experiments.

5. Conclusion

Networked personalised production arises when a group of manufacturing units operates independently, making autonomous decisions while supporting local engagement and global economic performance. The initial results encourage and validate the manufacturing unit simulator, pretending to have closer interactions with local suppliers and consumer hubs. The PPO-based agent was trained to accept demands and respond effectively to variations in the probability distribution of generated delays, primarily influenced by the quantity and complexity of product orders. Once trained, the reinforcement learning-based agent demonstrates faster response times when compared to a standard offline optimiser that would require multiple minor optimisations for each relevant sub-window within the planning horizon. Future research in reinforcement learning entails modelling a complete manufacturing network and considering cooperative and competitive relationships between agents, especially in scenarios with limited production resources.

Acknowledgements

This work was supported by Marie Skłodowska-Curie grant 847.577 (SMART4.0) and by grants from Science Foundation Ireland (SFI) 16/RC/3918 (CONFIRM), 12/RC/2289-P2 (Insight), co-funded by the European Regional Development Fund, and CAPES/BRAZIL (Finance Code 001).

Referências

Azarakhsh, S., Sahebi, H., e Seyed Hosseini, S. M. (2021). Design of a sustainable integrated crude oil manufacturing network with risk cover and uncertainty considerations: a case study. *Journal of Ambient Intelligence and Humanized Computing*, p. 1–14.

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

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., e Zaremba, W. (2016). Openai gym. *arXiv preprint arXiv:1606.01540*.

Chang, J., Yu, D., Hu, Y., He, W., e Yu, H. (2022). Deep reinforcement learning for dynamic flexible job shop scheduling with random job arrival. *Processes*, 10(4):760.

Cunha, B., Madureira, A. M., Fonseca, B., e Coelho, D. (2018). Deep reinforcement learning as a job shop scheduling solver: A literature review. In *International Conference on Hybrid Intelligent Systems*, p. 350–359. Springer.

Heckmann, I., Comes, T., e Nickel, S. (2015). A critical review on supply chain risk–definition, measure and modeling. *Omega*, 52:119–132.

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

Nachtigall, T., Mironcika, S., Tomico, O., e Feijs, L. (2020). Designing ultra-personalized product service systems. *CoDesign*, 16(4):274–292.

DOI: 10.59254/sbpo-2023-175124
http://live-proceedings.pantheonsite.io/p/175124

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

Qin, Z. e Lu, Y. (2021). Self-organizing manufacturing network: A paradigm towards smart manufacturing in mass personalization. *Journal of Manufacturing Systems*, 60:35–47.

Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., e Dormann, N. (2021). Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8. URL `http://jmlr.org/papers/v22/20-1364.html`.

Rauch, E., Dallasega, P., e Matt, D. T. (2017). Distributed manufacturing network models of smart and agile mini-factories. *International Journal of Agile Systems and Management*, 10(3-4):185–205.

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

Silbermayr, L. e Minner, S. (2014). A multiple sourcing inventory model under disruption risk. *International Journal of Production Economics*, 149:37–46. ISSN 0925-5273. The Economics of Industrial Production.

Strøm, M. A., Rehn, C. F., Pettersen, S. S., Erikstad, S. O., Asbjørnslett, B. E., e Brett, P. O. (2018). Combining design and strategy in offshore shipping. In *Marine Design XIII*, p. 147–162. CRC Press.

Sutton, R. S. e Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Tang, J. e Salonitis, K. (2021). A deep reinforcement learning based scheduling policy for reconfigurable manufacturing systems. *Procedia CIRP*, 103:1–7. ISSN 2212-8271. 9th CIRP Global Web Conference – Sustainable, resilient, and agile manufacturing and service operations : Lessons from COVID-19.

Tiihonen, J. e Felfernig, A. (2017). An introduction to personalization and mass customization. *Journal of Intelligent Information Systems*, 49(1):1–7.

Wang, H., Cheng, J., Liu, C., Zhang, Y., Hu, S., e Chen, L. (2022). Multi-objective reinforcement learning framework for dynamic flexible job shop scheduling problem with uncertain events. *Applied Soft Computing*, 131:109717. ISSN 1568-4946.

Zhang, Y., Zhu, H., Tang, D., Zhou, T., e Gui, Y. (2022). Dynamic job shop scheduling based on deep reinforcement learning for multi-agent manufacturing systems. *Robotics and Computer-Integrated Manufacturing*, 78:102412.

Zheng, P., Xia, L., Li, C., Li, X., e Liu, B. (2021). Towards self-x cognitive manufacturing network: An industrial knowledge graph-based multi-agent reinforcement learning approach. *Journal of Manufacturing Systems*, 61:16–26. ISSN 0278-6125.

Zhou, T., Zhu, H., Tang, D., Liu, C., Cai, Q., Shi, W., e Gui, Y. (2022). Reinforcement learning for online optimization of job-shop scheduling in a smart manufacturing factory. *Advances in Mechanical Engineering*, 14(3):16878132221086120.

DOI: 10.59254/sbpo-2023-175124
http://live-proceedings.pantheonsite.io/p/175124