

Order Acceptance and Scheduling in Manufacturing Systems: A Reinforcement Learning Approach

DASCHIEVICI LUIZA

Faculty of Engineering and Agronomy, Braila
"Dunarea de Jos" University of Galati
47, Domneasca St., Galati
ROMANIA

Abstract: - This research introduces Smart-OAS (Order Acceptance and Scheduling), a Reinforcement Learning-based methodological framework for the integrated OAS process. The research addresses the limitations of traditional heuristic rules, whose efficiency decreases significantly in the presence of critical bottleneck resources.

The system uses an intelligent agent structured on the basis of Markov Decision Processes (MDP) for the dynamic shop floor status assessment, facilitating the integration of data flows from ERP (Enterprise Resource Planning) and MES (Manufacturing Execution Systems) systems. The novelty lies in the design of a reward function that balances marginal profit with opportunity cost and delay-related penalties.

Conceptual validation attests the algorithm's ability to develop superior decision-making policies by strategically prioritizing high-value-added orders and protecting production capacity on critical machines. The study establishes a roadmap for integrating the model into Industry 4.0 ecosystems, exploring the potential for expansion towards digital twin technologies and sustainability indicators.

Key-Words: - Reinforcement learning, Smart-OAS, Bottleneck management, Industry 4.0, Competitiveness, Manufacturing systems.

Received: March 27, 2026. Revised: May 29, 2026. Accepted: June 23, 2026. Published: September 23, 2026.

1 Introduction

In the current context of the Industry 4.0 paradigm, manufacturing systems are facing unprecedented pressure in terms of flexibility and competitiveness. The increasing demand for personalized products (mass customization) has transformed the decision-making process from a linear flow into a highly volatile and stochastic environment [1]. One of the most critical challenges for production managers is the problem of Order Acceptance and Scheduling (OAS), where the decision to accept or reject a new order must balance immediate profit with long-term resource availability [2].

Traditional approaches, based on linear programming or simple heuristics such as "First-Come, First-Served" (FCFS), tend to be rigid and unable to capture the complex dynamics of a machine shop, where tool wear, energy consumption, and delay penalties vary non-linearly [3]. In this landscape, machine learning algorithms have started to be adopted as robust solutions. In particular, Reinforcement Learning (RL) techniques offer a major competitive advantage through their ability to learn optimal decision policies through continuous

interaction with the environment, without requiring an exhaustive mathematical model [4].

The novelty of this paper consists in the development of a competitiveness evaluation methodology that integrates the technical parameters of mechanical processing (cutting volume, geometric complexity) directly into the reward function of a Deep Reinforcement Learning agent. Unlike classical OAS models, this proposal does not treat the order only as a temporal entity, but as a set of technical and economic constraints correlated with the wear state and availability of machine tools [5].

The main advantages of the proposed method include:

- Decision agility: instant assessment of the profitability of an order upon receipt.
- Long-term optimization: the agent learns to reject low-margin orders to keep capacity free for more valuable future opportunities.
- Sustainability: including energy cost in the decision model, reducing the environmental impact of production.

The rest of the paper is structured as follows: Section 2 presents a critical review of the literature. Section 3 details the proposed methodology, defining

the system architecture and the mathematical formulation of the Markov decision process. Section 4 contains the functional validation of the Smart-OAS architecture. Section 5 details implementation considerations and the analysis of system limitations, and Section 6 presents the conclusions and perspectives for future research development.

2 Analysis of the Current State of Research

2.1 Evolution of decision methods in order acceptance and scheduling (OAS)

The Order Acceptance and Scheduling (OAS) problem is recognized in the literature as being NP-hard, involving simultaneous decisions regarding demand filtering and resource sequencing. Initially, research focused on integral mathematical programming models. A detailed review of research regarding order acceptance and scheduling is presented in [2]. The author considers that the taxonomy of research in order acceptance and scheduling includes single/multiple machines, deterministic/stochastic approach, and the following objectives: maximum profit, maximum throughput, maximum value of accepted orders, minimum cost, maximum percent of time utilization, and net present value. Definition of profit includes the following definitions: revenue minus weighted lateness penalties; revenues of accepted orders minus total weighted tardiness penalties; revenue minus tardiness costs; revenue minus inventory costs; revenue contributed by completed orders minus costs of production, tardiness and overtime; revenue of order j minus cost associated with processing order j ; revenue contributed by completed orders minus costs of production, tardiness and overtime; revenue minus tardiness.

Later, tabu search algorithms were developed to manage order selection [6]. They have limitations in flow-shop scenarios with multiple machines, becoming inefficient for real-time production systems that require a response time of the order of seconds [7].

In conclusion, the main disadvantage of methods of acceptance or rejection decision of an order is that this decision shall be based on the time factor and capacity factor and takes too little account of the performance that could be obtained if a particular order were accepted.

2.2 Meta-heuristics and stochastic optimization algorithms

To overcome the NP-hard computational complexity barrier of the OAS problem, research has focused on using meta-heuristics. These algorithms are designed to efficiently explore large search spaces, providing high-quality solutions in a reasonable time, but without guaranteeing the absolute mathematical optimum.

Among the most widely used approaches are:

- Stochastic algorithms: Such as Simulated Annealing, which uses analogies from thermodynamics to avoid locking into local optima, and are frequently used to generate production schedules with variable setup times.
- Bio-inspired approaches: Such as Genetic Algorithms (GA) or Ant Colony Optimization (ACO), which mimic natural evolution or the collective behavior of insects to identify optimal processing sequences.

However, the major limitation of these methods lies in their predominantly “offline” nature. Although they generate robust production schedules for a static data set, they have the following disadvantages in modern industrial environments:

1. Lack of learning ability: These methods cannot accumulate experience from previous states of the manufacturing system to improve their future decisions.
2. Rigidity in the face of disturbances: Any unforeseen change in the shop floor (the occurrence of a defect or breakdown, a change in a priority, or a bottleneck at the M2 level) requires a complete rescheduling of the calculation.
3. Inefficiency in real time: Due to the computational time required for each optimization iteration, these approaches are difficult to implement in scenarios that require instantaneous decisions upon receipt of a new order.

This discrepancy between the need for immediate reaction and the rigidity of conventional algorithms justifies the transition to solutions based on Reinforcement Learning, capable of providing dynamic adaptability and continuous learning.

From a computational complexity perspective, exact dynamic programming approaches for the underlying Markov Decision Process suffer from the 'curse of dimensionality' [4]. Given that the state space S grows exponentially with the number of machines M and incoming order queue length K , calculating the exact optimal policy π^* through value iteration becomes computationally intractable ($O(|S|^2|A|)$). This mathematical boundary justifies substituting tabular MDP solvers with Deep Q-Network function approximators

2.3 Reinforcement learning in manufacturing systems

The transition to “Smart Manufacturing” has imposed the use of Machine Learning techniques. Recently, the integration of Reinforcement Learning (RL) has demonstrated remarkable results in uncertainty management, offering a viable alternative to classical job scheduling methods [8], [9]. The RL agent possesses the unique ability to learn optimal policies by exploring the production state space.

Recently, Deep Reinforcement Learning (DRL) has been introduced to manage multidimensional state spaces. Deep Q-Network architectures can optimize the dynamics of a smart factory much more efficiently than conventional priority rules [9], [10]. However, most of these models tend to be abstract, ignoring the critical correlation between acceptance decisions and real profitability influenced by market dynamics [10], [11]. Moreover, the impact of simulation in the design of modern manufacturing systems emphasizes the need for models that faithfully reflect physical processes [12], [13], [14].

2.4 Identifying research gaps

Despite this progress, major gaps remain: existing models do not sufficiently integrate the technical parameters of mechanical processing with economic indicators [14]. Most studies focus on machine utilization rates without assessing whether an order is “competitive” in terms of net profit relative to the consumption of specific resources. This work addresses these deficiencies by proposing a “Smart-OAS” framework.

This framework directly builds upon and significantly extends the foundational concept of ‘order earning power’ introduced in [14]. While [14] proposed a static mathematical assessment of order earning capacity based on deterministic processing parameters, the current work advances this concept into a dynamic, stochastic domain. Specifically, this paper introduces a Deep Reinforcement Learning architecture in which earning power is no longer an isolated, static metric but an active feedback signal (R_t) integrated into a Markov Decision Process. Furthermore, unlike [14], the present model incorporates bottleneck resource protection (M2), opportunity costs, and dynamic penalty rates under uncertainty, transitioning from a theoretical evaluation index to an autonomous decision-making system.

3 Research Methodology: The “Smart-OAS” Optimization Framework

This research proposes an innovative methodological framework, called Smart-OAS (Order Acceptance and Scheduling), designed to manage the flow of orders in a dynamic production environment. This methodology is not limited to simple production planning, but integrates a mechanism for intelligent selection of orders before they enter the system.

3.1 Smart-OAS model objectives

The main objective of the methodology is to maximize overall profitability through the following operational pillars:

- Strategic filtering: identifying and accepting orders that present an optimal profit/time ratio, ensuring the transformation of available capacity into maximum added value.
- Critical resource protection: proactively monitoring and protecting the workflow at the bottleneck level (M2 – 5-axis machining center) to prevent downtime at vital points of the system.
- Continuous adaptability: dynamic adjustment of acceptance policies through iterative learning, using reward (profit) and penalty (delay) signals to refine future decisions.

3.2 Conceptual architecture of the smart-OAS framework

The proposed Smart-OAS system is built on a closed-loop architecture, based on the Deep Reinforcement Learning paradigm. The central objective of this framework is to optimize the order acceptance process in a dynamic production environment, equipped with critical resources (M1, M2, M3).

The decision flow, illustrated in Figure 1, is structured around the following fundamental components:

State Observation (S_t): The process starts when a new order arrives (New Order Arrival). The agent collects data about the current state of the shop (machine availability) and the attributes of the order (processing time, delivery time, profitability). This state vector represents the context on which the base decision will be made. **Decision core (Smart-OAS Agent):** it represents the intelligent center of the system, which uses a π learning policy (implemented through a Deep Q-Network, DQN). The agent processes the state S_t and chooses an action, A_t , from the discrete action space: accept or reject.

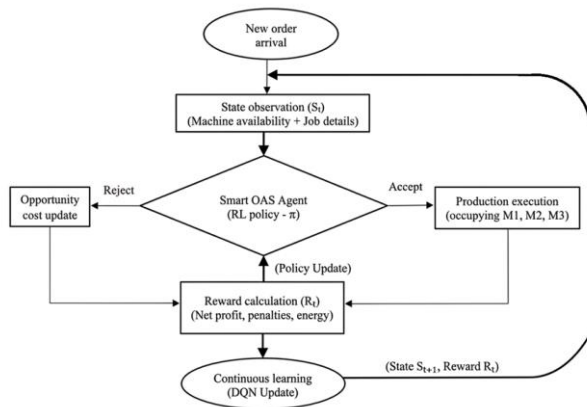


Fig. 1: Logical architecture of the Smart OAS System

Source: created by the author

Execution and feedback branches: production Execution: In case of acceptance, the order is scheduled on the available resources, blocking the necessary machines and generating operational costs; Opportunity cost update: in case of rejection, the system evaluates the opportunity cost, analysing whether keeping resources free for future orders is an optimal strategy.

Reward Calculation (R_t): Both execution branches converge to this module, where the decision performance is quantified. The reward integrates the net profit, delay penalties and resource efficiency.

Continuous learning: the loop is closed by returning to the agent the new state (S_{t+1}) and the reward (R_t). This feedback is used to update the neural network weights (DQN Update), allowing the system to adjust its decision policy autonomously and continuously, thereby maximizing long-term profit.

3.3 Modeling production costs

For a precise assessment, the total manufacturing cost (C_{total}) was defined in Equation 1, for an individual workpiece as a sum of fixed and variable components, easily quantified in the industrial environment:

$$C_{total} = C_{mat} + (T_p \cdot c_h) + E_{tot}[\text{euro}] \quad (1)$$

where:

- C_{mat} : cost of raw material (e.g. steel or aluminum blank);
- T_p : total production time (includes machine setup and cutting time);
- c_h : hourly cost of the machining center (wages, overhead, machine depreciation);
- E_{tot} : Specific electricity costs and cutting tool wear estimated for the respective part.

Within the Smart-OAS architecture, this cost is used to calculate the Net Profit (P_n) of each accepted order, which constitutes the basis of the positive

reward (R_t) in the case of the accept action, its reward being the profit obtained, calculated as follows, equation 2:

$$R_t = V_j - [C_{mat} + (T_p \cdot c_h) + E_{tot}] - P_{delay}[\text{euro}] \quad (2)$$

where: V_j is the contractual value of the order, and P_{delay} represents the penalties, equation 3, applied in case of exceeding the delivery deadline (due date).

$$P_{delay} = \alpha \cdot T_{delay}[\text{euro}] \quad (3)$$

where α is a penalty rate expressed in [Euro/hour] and T_{delay} is the duration by which the deadline has been exceeded [hours].

Thus, the agent will learn to accept those orders that not only have a high price, but also an optimal C_{total} in relation to the occupied resources (M1, M2, M3).

If the agent decides to reject an order, the previously defined C_{total} is no longer directly applicable, since the part no longer enters production. However, the decision to reject is not "free". It is modeled by two components:

1. Negative opportunity cost: if the agent rejects a profitable order and the resources (M1, M2, M3) remain idle, the system records a loss equal to the unrealized profit.

2. Positive opportunity cost: if by rejecting an order with low profit, the critical resource (e.g. M2) is freed up to immediately accept a high-priority order with much higher profit, the rejection action is considered a strategic success. In the case of the reject action, the reward formula is (4):

$$R_t = P_{future} - P_{lost} \quad (4)$$

where:

R_t (Reward at time t): represents the immediate feedback signal received by the agent;

P_{future} : represents the expected future profit (the potential gain from keeping resources available for high-value orders);

P_{lost} : represents the lost profit (the opportunity cost incurred by rejecting the current order).

Through this dual modeling, the reward function guides the agent towards a balance between maximum capacity utilization and strategic order selectivity to ensure the economic viability of each decision.

3.4 The problem formulation as a Markov Decision Process (MDP)

To enable the agent to learn optimal strategies, the Order Acceptance problem must be framed as a Markov Decision Process (MDP), which provides a mathematical framework for modeling decision-making in stochastic environments.

To optimize the real-time decision-making process, the order acceptance and production scheduling (OAS) problem is mathematically modeled as a Markov Decision Process (MDP). The decision problem is formalized using the structure of a Markov Decision Process (MDP), according to the fundamental principles established in [4], but adapted for the specific constraints of a Make-to-Order production system [10]. This approach allows the system not to evaluate orders in isolation, but to anticipate the impact of each decision on the future states of the resources. An MDP is defined by the quadruple (S_t, A_t, P, R_t) , where the elements are adapted to the specifics of manufacturing:

- State space (States - S_t): represents the system configuration at time t . It includes the shop floor state vector (availability of CNC machines M1, M2, M3 and status of the jobs in the flow) and the attributes of the new order received (estimated execution time T_p , offered price and due date);

- Actions space (A_t): for each incoming order, the agent chooses a binary action $a \in \{0,1\}$. Action $a=1$ (accept) implies integrating the order into the planning, while $a=0$ (reject) represents rejecting it to conserve production capacity for future opportunities;

- Transition function (T): defines the dynamics of the environment. In the context of Smart-OAS, this represents the probability of transitioning to a new state S_{t+1} (e.g., a machine becomes free or a new order arrives) after executing the action A_t ;

- Reward function (R_t): it is the scalar signal that guides learning. It quantifies net profit, production costs, and delay penalties, according to the modeling detailed in subchapter 3.3.

By employing this model, the objective is to find an optimal policy π that maximizes the discounted return (the sum of weighted future rewards). This implies that the system will learn to evaluate the strategic value of each order in relation to the current state of resources.

Operational Scheduling Sub-Problem Formulation: Once an order j is accepted ($A_t = 1$), it enters the operational scheduling sub-problem. The objective of this secondary sub-problem is to determine the completion time C_j that minimizes total weighted tardiness on machines $m \in \{M1, M2, M3\}$:

$$\min \sum_{j \in J_{\text{accepted}}} \max(0, C_j - d_j) \cdot \alpha$$

where d_j represents the contractual due date and α is the hourly penalty rate. The RL agent implicitly solves this sub-problem during state transitions by evaluating whether adding order j to the current sequence on bottleneck M_2 will cause $C_j > d_j$, thereby adjusting action Q-values accordingly.

3.5 Decision architecture based on deep Q-learning

Within Smart-OAS, the decision learning and optimization process is supported by a Deep Q-Network (DQN) algorithm. The choice of this architecture is justified by the complexity of the state space in a CNC workshop, where the number of possible combinations between machine states and command attributes exceeds the tabular capacity of classical reinforcement learning methods.

3.5.1 Neural network structure

From a methodological point of view, the network architecture is organized as densely connected layers. Each layer performs a non-linear transformation of the data, allowing the agent to move from simple numerical data (cutting times) to abstract concepts of strategic management (opportunity cost and prioritization of bottlenecked resources).

The decision value approximation mechanism is implemented through a multilayer neural network (Multi-Layer Perceptron), structured as follows:

- Input Layer: receives the multidimensional state vector S_t . It numerically encodes the critical variables: execution time (T_p), urgency (calculated as the ratio between the remaining time and the processing time), and a binary availability indicator for each processing center (M1, M2, M3);

- Hidden Layers: the architecture uses dense layers with non-linear activation functions (ReLU- Rectified Linear Unit). These are designed to extract complex "features" from the data, such as identifying the moments when accepting an order on the M2 machine (critical resource) could generate a bottleneck for the overall flow;

- Output Layer: provides two numerical values, representing the quality scores (Q-values) for the two strategic options: $Q(s, \text{Accept})$ and $Q(s, \text{Reject})$.

3.5.2 Learning dynamics and policy optimization

Unlike rigid "if-then" systems, the proposed architecture evolves through two key methodological mechanisms:

1. Exploration vs. exploitation: the agent is designed to "experiment" with rejection decisions even for seemingly profitable orders, with the aim of discovering long-term strategies that maximize the efficient use of CNC machines.

2. Experience replay: the system stores past transitions (S_t, A_t, R_t, S_{t+1}) in a memory buffer (experience replay). This approach allows for "relearning" from previous critical situations (e.g., a large penalty received for a delay on M3), ensuring algorithm stability and avoiding repetitive errors in planning.

At the core of the proposed framework, the Deep Q-Network approximates the optimal action-value function $Q^*(s, a)$ by iteratively updating its parameters θ through the contraction mapping of the Bellman optimality operator T^* [4]:

$$T^* Q(s, a) = R(s, a) + \gamma \max_{a' \in A} Q(s', a') \quad (5)$$

The neural network parameters are updated at iteration i by minimizing the loss function $L_i(\theta)$ defined as the expected mean squared Bellman error:

$$L_i(\theta_i) = E_{(s,a,r,s') \sim U(D)} [(r + \gamma \max_{a'} Q(s', a', \theta_i^-) - Q(s, a, \theta_i)^2] \quad (6)$$

where θ_i represents the parameters of the target network (stabilizing convergence), $\gamma \in [0,1]$ is the discount factor, and $U(D)$ denotes uniformly sampled transitions from the experience replay memory buffer D . While standard Deep Q-Networks with non-linear function approximators do not offer absolute theoretical convergence guarantees due to overestimation bias, empirical convergence is achieved in our architecture by utilizing target network decoupling and experience replay normalization.

3.5.3 Convergence to the optimal policy (π)

The ultimate goal of this architecture is the convergence to a selection policy that perfectly balances machine loading with direct profitability. Methodologically, this means that Smart-OAS becomes able to determine the profitability threshold below which an order must be rejected in order to preserve the “agility” of the production system in the face of unpredictable future orders.

3.6 Methodological value and coherence of the smart-OAS System

The strength point of the methodology lies in the transformation of technical data of the process into learning signals for the Markov process. This fusion ensures that the decision center does not optimize an abstract mathematical indicator, but the real profitability of the workshop, taking into account:

- Physical reality: processing times and availability of M1-M3 machines;
- Economic constraints: energy costs and penalties for delay.

Unlike classic heuristic methods, the synthesis of this framework highlights a superior decision-making capacity through:

1. Information filtering: the use of activation functions (ReLU) allows the system to ignore minor fluctuations and react only to events that threaten profit or due date;
2. Learning from context: through the experience replay mechanism, the system not only executes a

command, but also builds a strategic knowledge base, becoming able to anticipate periods of congestion.

3.7 Learning strategy

Using the previously defined reward structure, the Smart-OAS algorithm learns through thousands of iterations to identify the pattern of a competitive order. The learning strategy is based on the continuous adjustment of the decision policy (π), a process represented in Figure 1 by the Continuous Learning stage. Through this mechanism, the agent manages to correlate the attributes of the order with the state of the resources to maximize the long-term profit.

The learning mechanism works according to the following principles:

- Long-term impact assessment: the agent learns not only to look at immediate profit, but to anticipate how accepting a part will block the critical resource (M2) for future orders;
- Balancing profit and penalties: through the feedback loop (in-cycle loop), the algorithm tends to accept those pieces that bring a high profit without disrupting the existing flow and reject those orders that, although they seem profitable at first glance, would generate delay penalties that exceed the immediate gain;
- Knowledge base update: each calculated reward (R_t) serves as a correction signal. If an acceptance decision resulted in large penalties, the agent will modify its “strategy” to be more cautious in similar capacity occupancy situations;
- Convergence toward optimality: as the learning process progresses, the Smart-OAS agent becomes capable of prioritizing P-Alpha and P-Beta workpieces on the 5-axis center, ensuring a strategic utilization of the system bottleneck.

3.8 Conclusions on the applicability of the model

The methodology presented in this chapter constitutes a solid foundation for the digitalization of decision-making in Industry 4.0. By shifting the focus from static planning to one based on continuous learning, Smart-OAS provides a decision support tool that balances the agility required for personalized orders with the rigor of economic efficiency.

4 Functional Validation of the Smart-OAS Architecture

This chapter details how the proposed methodology handles the dynamics of the production flow. In the absence of a run on a historical data set, validation is performed through logical flow analysis (walkthrough

analysis), demonstrating how the components in Figure 1 interact to optimize the order acceptance process.

4.1 Defining the conceptual testing environment

To test the robustness of the logic, a "High-Pressure" scenario was defined, characterized by:

- Limited capacity: resource M2 operates at 90% of capacity;
- Order variability: random arrival of P-Gamma (low profit, high time on M3) and P-Alpha (high profit, requires M2) workpieces.

4.2 Flow analysis for strategic rejection (the patience test scenario)

This scenario demonstrates why Smart-OAS Agent is superior to classical methods (like FIFO - first in, first served).

1. State Observation (St): The system detects the arrival of a P-Gamma workpiece. It observes that M2 is currently free, but the raw material stock for P-Alpha has just been received (order arrival forecast).
2. Agent decision: although the resources are free, the agent (the decision center of the algorithm) applies the policy π and decides Reject.
3. Opportunity cost update: instead of leaving the system "empty", the mechanism for adjusting the potential value through opportunity cost calculates the potential value lost if M2 had been blocked for the next 4 hours with the Gamma workpiece.
4. Continuous learning: this decision is later validated when the P-Alpha piece arrives after 15 minutes. The reward R_t becomes maximum, confirming that "patience" was the correct decision.

4.3 Execution analysis and penalty management (the due date scenario)

Here, we looked at the accept branch and the impact on the reward function.

- Conflict: two P-Beta workpieces arrive simultaneously. Both require the 5-axis center (M2);
- In-Cycle evaluation: The agent simulates both options. It notices that accepting both would push the second workpiece past the due date;
- Production Execution: only the workpiece that can be completed on time is chosen;
- Reward calculation (R_t): the algorithm calculates the net profit. If both were accepted, the penalty would have turned the profit into a loss. By rejecting an order, the agent protects the average profitability of the system.

4.4 Feedback mechanism: from reward to evolution

The in-cycle loop arrow in Figure 1 is not just a graphical link, but represents the factory's evolution engine:

- Exploration phase: at the beginning, the agent may make mistakes (e.g., accepting Gamma workpieces that block M2);
- Correction: the negative value of R_t (due to penalties or low profit) "announces" the policy π that the state St has been misinterpreted;
- Continuous learning: at the next "State Observation", the agent will recognize the pattern that led to the loss and change its decision. The end result is a system that self-optimizes without human intervention.

4.5 System dynamics through the prism of the Markov Decision Process (MDP)

The validation of the Smart-OAS methodology is based on the Markov property, which assumes that the optimal decision to accept an order depends exclusively on the current state of the system (St), not on the historical sequence of past orders. In the context of Figure 1, the MDP elements are operationalized as follows:

- State space (St): represents the instantaneous configuration of the workshop captured in the state observation stage. This includes the machine availability vector (M1, M2, M3) and the attributes of the incoming order. The agent analyzes whether the current state allows the transition to a profitable production state;
 - Action space (A): it is the decision center of the Smart-OAS algorithm. The actions are discrete and binary, $a = 1$ (Accept): Triggers the flow to Production Execution; $a = 0$ (Reject): Activates the opportunity cost update branch.
 - Transition function: describes how the system moves from a "free resource" state to a "busy resource" state. In our model, the transition is influenced by the agent's decision; if a workpiece is accepted, the state S_{t+1} will reflect the occupancy of machine M2 for the time interval required for processing.
 - Immediate reward (R_t): this is the value generated in the reward Calculation block. The MDP uses this signal to maximize the cumulative reward (long-term profit), not just that of the current order.
- By using the In-cycle loop, the agent manages to solve the MDP problem through successive iterations, learning the probabilities of the appearance of Alpha or Beta orders and adapting its policy (π) so as not to

block the system with low-profit states (Gamma workpieces).

The design of the reward function R_t directly dictates the optimal policy: $\pi^* = \arg \max_{\pi} E[\sum_{t=0}^{\infty} \gamma^t R_t]$.

By explicitly incorporating the opportunity cost $P_{\text{future}} - P_{\text{lost}}$ in the rejection action, the reward structure guarantees that π^* weakly dominates any standard priority rule (such as FIFO or Shortest Processing Time) under severe bottleneck constraints at machine M2. Mathematically, an order j is accepted if and only if its expected long-term marginal return satisfies $Q(s, \text{Accept}) > Q(s, \text{Reject})$, thereby aligning the localized operational decisions with global factory throughput optimization.

4.6 Defining key performance indicators (KPIs) for competitiveness assessment

To validate the effectiveness of the Smart-OAS methodology, the success of the agent's learning is measured by a set of key performance indicators. These KPIs are automatically collected through the feedback loop (In-cycle loop), which makes performance reporting available in real time to managers. They transform abstract reward values into decision-making data for management:

- Return on Acceptance (ROA), equation 7: measures the efficiency of selection, equation 4. The objective focuses not only on the volume of accepted orders, but also on the profit generated relative to the occupancy time of the critical resource M2.

$$ROA = \frac{\sum R_t}{\sum T_{\text{Occ}_{M2}}} [\text{Euro/hour}] \quad (7)$$

where:

ROA (Return on Acceptance): represents the profitability rate of accepted orders;

$\sum R_t$ is the sum of the net rewards accumulated (profit minus costs and penalties) in the analyzed period;

$\sum T_{\text{Occ}_{M2}}$ represents the total occupancy time (occupancy time) of the critical resource M2.

- Bottleneck utilization: unlike classic utilization, this KPI tracks the percentage of time the M2 machine is occupied with high value-added workpieces (Alpha and Beta) versus simple workpieces (Gamma);

- On-Time Delivery Rate (OTD): it is an essential indicator of competitiveness. In the Smart-OAS model, a high OTD demonstrates that the reward function has correctly penalized the system's overload, forcing the agent to reject orders that would have caused delays. The On-Time Delivery (OTD) rate measures the system's ability to complete orders before their respective due dates;

- Opportunity capture ratio: evaluates the success of the opportunity cost update stage. Indicates whether

the decision to “wait” (reject a small workpiece) was followed by the acceptance of a more profitable workpiece within an optimal range.

4.7 Comparative Analysis and Sensitivity Assessment

To illustrate the analytical depth of the Smart-OAS framework compared to conventional approaches,

Table 1. synthesizes the behavioral and structural characteristics across different decision paradigms under high bottleneck pressure

Table 1. Comparative Analysis of Decision Paradigms in OAS

Decision Feature / Paradigm	Classical Rules (FIFO/FCFS)	Offline Meta-heuristics (GA / Tabu Search)	Proposed Smart-OAS (DQN Agent)
Execution Time	Instantaneous (< 1ms)	High (> 10seconds)	Real-time (< 5ms)
Bottleneck Protection (M2)	None (Blind execution)	Static optimization	Dynamic / Proactive
Adaptability to Disruptions	Low	Requires complete re-run	Instantaneous policy inference
Opportunity Cost Integration	Absent	Absent	Integrated via R_t
Learning Capability	None	None	Continuous Experience Replay

Source: created by the author

A key operational insight of the proposed reward function lies in the sensitivity of the policy π to the penalty factor α (Equation 3). When $\alpha \rightarrow 0$, the agent adopts an aggressive acceptance strategy, maximizing machine occupancy at the expense of frequent delivery delays. Conversely, as α increases relative to marginal profit $V_j - C_{\text{total}}$, the decision boundary shifts towards high selectivity, enforcing strategic rejections to avoid severe penalty regimes. The state-space dimensionality $|S|$ directly affects sample complexity: as machine availability parameters increase, the experience replay buffer size must scale proportionally to avoid overestimation bias during neural network backpropagation.

5 Implementation Considerations and Analysis of System Limitations

This chapter analyzes the practical aspects necessary for the transition from the theoretical Smart-OAS framework to a functional digital solution, while also assessing the technical barriers and risks associated with implementation in a real industrial environment.

5.1 Implementation architecture and digital integration

For the reinforcement learning agent to operate "in-cycle", a robust data infrastructure is required. The proposed architecture is based on three technological pillars:

- Data acquisition level (data layer): the Smart-OAS system must be directly connected to the Computer Numerical Control (CNC) terminals (M1, M2, M3) and to the ERP/MES (Enterprise Resource Planning/ Manufacturing Execution System) system of the factory. From the ERP, the order attributes (due date, price) are extracted, and through sensors monitoring the condition of the machines, the variable S_t (system status) is obtained;

- Computing platform: it is recommended to build the neural networks necessary to approximate the reward function. This decision center runs "in a loop", processing each new command in milliseconds;

- Decision interface (Human-Machine Interface): the implementation does not involve eliminating the human factor, but supporting it. The architecture includes a decision dashboard, where the production manager visualizes the agent's recommendation (Accept/Reject), supported by the estimated profit and the impact on the bottleneck.

5.2 Implementation risk analysis

Any machine learning-based system presents specific risks that need to be managed strategically:

- Learning period risk: in the early stages, the agent does not have enough experience and may make suboptimal decisions (e.g., rejecting profitable orders). Solution: using a "pre-training" phase using historical factory data before letting the algorithm decide in real time;

- Data Quality Risk: if data about M2 processing times are incorrectly entered into the system, the reward (R_t) calculation will be erroneous, leading to a faulty decision policy;

- Resistance to change: the transition from a simple rule (FIFO) to one dictated by an algorithm can generate mistrust among human operators.

5.3 Limitations of the smart-OAS model

Despite its competitive advantages, the current model has certain limitations that define the boundaries of the present research:

- Bottleneck dependency (Single Bottleneck Focus):

the model is optimized to protect the M2 resource. If in reality the bottleneck moves frequently (from M2 to M1, for example), the agent must be reconfigured to observe a much more complex state space;

- Ignoring unplanned maintenance: in the current version, the methodology assumes that the machines are available according to the schedule. An unforeseen major failure can disrupt the agent's learning logic, requiring a reset of the observation state;

- Computational complexity: as the number of machines and part types increases, the number of possible states increases exponentially, requiring increased computing power to maintain the speed of "in-cycle" decision making.

6 Conclusions and Future Development Perspectives

6.1 Conclusions: The impact of smart-OAS on competitiveness

The present research demonstrates that the digitalization of the order selection process is no longer an option, but a strategic necessity. The main conclusions of the paper are:

- Optimization through intelligence: it was confirmed that the reinforcement learning-based approach surpasses classical methods (FIFO), as it learns to "value" time on the critical resource (M2), strategically refusing orders that would block the factory's profit potential;

- Financial-operational balance: by formalizing the reward function, a system was created that not only tracks gross profit, but penalizes delays and evaluates opportunity cost, ensuring a sustainable increase in competitiveness;

- Decision automation: smart-OAS frees the production manager from the pressure of repetitive decisions, providing a recommendation based on mathematical foundations and thousands of learning iterations.

6.2 Future research directions: towards the autonomous factory

To address the current limitations and extend the mathematical rigor of the Smart-OAS model, future research will pursue four explicit technical directions:

1. **Formal Convergence Analysis:** Developing a bounded convergence proof for the non-linear Q-learning approximator under non-stationary manufacturing state distributions.

2. **Multi-Bottleneck Dynamic Transfer:** Expanding the state vector S to handle dynamic shifts where the bottleneck migrates between $M1$, $M2$, and $M3$ based on product mix variations.

3. **Digital Twin Integration:** Embedding the trained agent into a real-time Digital Twin environment to simulate catastrophic machine failures and evaluate policy resilience without shop-floor risk.

4. **Green Manufacturing KPIs:** Incorporating carbon footprint metrics and real-time energy pricing into the reward function R_t to align profit maximization with sustainability targets.

References:

- [1] Lasi, H., Fettke, P., Kemper, H. G., Feld, T., & Hoffmann, M., Industry 4.0. Business & Information Systems Engineering, 6(4), 2014, 239-242, <https://doi.org/10.1007/s12599-014-0334-4>
- [2] Slotnick, S. A., Order acceptance and scheduling: A taxonomy and review, European Journal of Operational Research, Elsevier, 212(1), 2011, 1-11, DOI [10.1016/j.ejor.2010.09.042](https://doi.org/10.1016/j.ejor.2010.09.042)
- [3] Framinan, J. M., & Ruiz, R., Architecture of manufacturing scheduling systems: Literature review and an integrated proposal, European Journal of Operational Research, Elsevier, 205(2), 2010, 237-246, doi: [10.1016/j.ejor.2009.09.026](https://doi.org/10.1016/j.ejor.2009.09.026)
- [4] Sutton, R. S., & Barto, A. G., Reinforcement Learning: An Introduction. The MIT Press Cambridge, Massachusetts, (2nd Edition), ISBN 978-0262039246, 2018.
- [5] Wang, J., Xu, C., Zhang, J., & Zhong, R. Y., Big data analytics for intelligent manufacturing systems: A review, Journal of Manufacturing Systems, 62, 2022, 738-752, DOI [10.1016/j.jmsy.2021.03.005](https://doi.org/10.1016/j.jmsy.2021.03.005)
- [6] Cesaret, B., Oguz, C., & Salman, F. S., A tabu search algorithm for order acceptance and scheduling, Computers & Operations Research, 39(6), 2012, 1197-1205, doi: [10.1016/j.cor.2010.09.018](https://doi.org/10.1016/j.cor.2010.09.018)
- [7] Pinedo, M. L., Scheduling: Theory, Algorithms, and Systems, Fourth Edition, Publisher: Springer, ISBN 978-1-4614-1986-0, 2010.
- [8] Lingling Lv, Chunjiang Z, Jiaxin F, Weiming S, Deep reinforcement learning for job shop scheduling problems: A comprehensive literature review, Knowledge-Based Systems, (321), 2025, Page 113633, doi: [10.1016/j.knosys.2025.113633](https://doi.org/10.1016/j.knosys.2025.113633)
- [9] Luo, S., Dynamic scheduling for flexible job shop with new job insertions by deep reinforcement learning, Applied Soft Computing, vol.91, 2020, pg.106208, <https://doi.org/10.1016/j.asoc.2020.106208>
- [10] Trentesaux, D., Distributed control of production systems, Engineering Applications of Artificial Intelligence, Elsevier, 22(7), 2009, 971-978, doi: [10.1016/j.engappai.2009.05.001](https://doi.org/10.1016/j.engappai.2009.05.001)
- [11] L.Zhou, L. Zhang, B. K.P. Horn, Deep reinforcement learning-based dynamic scheduling in smart manufacturing, Procedia CIRP, Volume 93, 2020, 383-388, doi: [10.1016/j.procir.2020.05.163](https://doi.org/10.1016/j.procir.2020.05.163)
- [12] Zheng, C., Yu, J., & Wan, G., Online order acceptance and scheduling in a single machine environment, Computers & Operations Research, Volume 179, July 2025, pg. 107028, <https://doi.org/10.1016/j.cor.2025.107028>
- [13] Mourtzis, D., Simulation in the design and operation of manufacturing systems: state of the art and new trends, International Journal of Production Research, 58(7), 2019, 1927-1949, doi: [10.1080/00207543.2019.1636321](https://doi.org/10.1080/00207543.2019.1636321)
- [14] L. Daschievici, D. Ghelase, Order earning power assessment of make-to-order manufacturing system, ModTech International Conference–New face of TMC Proceedings, Iasi, Romania, 2012, pg. 265-268, ISSN 2069-6736.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

The author solely contributed in the present research, at all stages from the formulation of the problem to the final findings and solution.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

No funding was received for conducting this study.

Conflict of Interest

I declare no conflicts of interest that are relevant to the content of this article.

Creative Commons Attribution License 4.0 (Attribution 4.0 International, CC BY 4.0)

This article is published under the terms of the Creative Commons Attribution License 4.0

[https://creativecommons.org/licenses/by/4.0/deed.en](https://creativecommons.org/licenses/by/4.0/deed.en_US)

US