POLITECNICO
MILANO 1863

School of Industrial and Information Engineering

Master of Science in Management Engineering

Production scheduling based on a reinforcement learning algorithm for complex production systems

Advisor: **Prof. Luca Fumagalli**
Co-advisors: **Prof. Elisa Negri,**
Lorenzo Ragazzini

Authors:
Carmen Aurora Adanaque Infante
Student ID: 925700
Andrés Felipe Simbaqueba Sánchez
Student ID: 925268

Academic year 2020/2021

Acknowledgments

I want to thank God for bringing me this far for going further. To my parents and my siblings for their support in every decision I make. Thank you for believing in me even when I do not. Also, I want to give my gratitude to my academic advisors, Elisa Negri, and Lorenzo Ragazzini, for their support and guidance during the development of this work. Thanks for allowing me to work with you. Lastly, I want to thank my thesis co-worker, Carmen Adanaque, for the support she gave me, not only as a colleague but also as a friend.

Andrés Felipe Simbaqueba Sánchez

I want to thank God for giving me this moment. To my parents for being dedicated and perseverance. To my sister, for being admired and example. To Marcelo for being my support along all these years and my motivation to be a better person every day. I would also like to thank Elisa Negri and Lorenzo Ragazzini for the opportunity to work on this challenging topic and their guidance in this process. Finally, I want to thank my thesis co-worker Andres Simbaqueba for his passion for making things happen.

Carmen Aurora Adanaque Infante

Abstract (English version)

Today, the manufacturing sector is looking for complex production systems to satisfy its flexibility requirements without neglecting the trade-off between being at the forefront of markets' needs and being profitable. Due to the changes in those needs, the scheduling function becomes a relevant issue to solve by manufacturing companies. Furthermore, in the Industry 4.0 context, some emerging technologies as artificial intelligence facilitate several quotidian activities in a production plant. For example, most of the production planning and control activities present high variability, making decision-making difficult. However, through automatization, it is possible to simplify it. Moreover, AI approaches bring improvements not only in terms of performance but also in terms of cost.

Putting together the production scheduling problem related to complex production systems and the advantages that emerging technologies bring to solve the issues inside these environments, it is essential to develop an alternative way to solve the scheduling problem considering an innovative perspective. One of the options to explore is applying AI techniques, an approach that has not been explored enough and that presents potential features in different dimensions (i.e., accuracy, computational cost, precision, etc.)

In this thesis, we developed and implemented a tool to solve the scheduling problem of a manufacturing system based on a reinforcement learning (RL) method. Compared to a well-known dispatching rule (i.e., shortest processing time), the proposed contribution resulted in improvements for resource utilization and lower work in progress due to the

shortening of the flow time of the jobs. However, the computational cost is still a challenge to be faced in future research.

Abstract (Italian version)

Attualmente, nel settore manifatturo stanno sviluppando sistemi complessi di produzione per soddisfare la flessibilità nelle sue richieste, senza trascurare il compromesso tra essere all'avanguardia delle necessità dei mercati ed essere economicamente competitivo. Poiché i mercati cambiano frequentemente, la pianificazione è una questione prioritaria da risolvere nelle aziende di manifatture.

Inoltre, nel contesto della industria 4.0, alcune tecnologie emergenti come l'intelligenza artificiale semplifica alcune attività che si svolgono ogni giorno all'interno di un impianto. Per esempio, la maggior parte dei processi relativi alla pianificazione e al controllo della produzione presentano un'elevata variabilità, rendendo difficile il processo di prendere decisioni. In più, i miglioramenti che portano gli approcci in IA, non sono solo in termini di prestazioni, sono anche i termini di costi.

Pertanto, approfittare i vantaggi delle tecnologie emergenti nella risoluzione dei problemi di pianificazione all'interno dei sistemi complessi di produzione è possibile tramite lo sviluppo di soluzioni con una perspettiva innovativa. Si sono esplorate applicazioni di tecniche AI, un approccio che non è stato analizzato a sufficienza e che presenta caratteristiche potenzialmente favorevoli nei confronti della precisione, costo computazionale, etc.

Questa tesi si centra nello sviluppo e implementazione di un algoritmo di intelligenza artificiale basato sul metodo dell'apprendimento per rinforzo (RL). L'analisi attraverso le metriche di rendimento note, come il tempo impiegato nel processo, evidenzia

che il contributo proposto in questo lavoro garantisce miglioramenti in termini di utilizzo di risorse e di diminuzione del tempo di attraversamento dei prodotti. Questioni come il costo computazionale sono una sfida che deve ancora essere confrontata in ricerche future.

Table of Contents

List of Abbreviations ..... **xi**

List of Tables ..... **xiv**

List of Figures ..... **xv**

Executive Summary ..... **xvii**

Chapter 1 - Introduction **1**

Chapter 2 - Research Design **5**

2.1 Systematic literature review research objectives 5

2.2 Research methodology 6

2.2.1 Literature review gap examination 6

2.2.2 Objective definition 7

2.2.3 Theory critical examination 7

2.2.4 Development of the contribution 7

2.2.5 Discussion of the results, limitations, and further research 8

Chapter 3 - Systematic Literature Review Methodology **9**

3.1 Planning the review 9

3.1.1 Analytical framework 10

3.1.2 Performance improvement 11

3.1.3 System dimensions 13

3.2 Conducting the review 15

3.3 Reporting and dissemination 18

3.3.1 System features 19

3.3.2 Implementation features 20

3.3.3 Performance 21

3.4 Answer to research questions 22

3.5 Bibliometric analysis 26

3.5.2 Keyword's co-occurrence map 27

3.5.3 Use of constructs 28

3.5.4 Productivity per country 31
3.5.5 Co-authorship network 32

Chapter 4 - SLR Results Discussion 34

4.1 General discussion 34
4.2 Machine learning approach 36
4.2.1 Machine learning as indirect solution 37
4.2.2 Machine learning as direct solution 41
4.2.3 Remarks 46
4.3 Research objectives 46
4.3.1 Motivation 46
4.3.2 Objectives 49

Chapter 5 - Theoretical Background 52

5.1 Production scheduling 52
5.1.1 Notation 52
5.1.2 Production system environment 53
5.1.3 Constraints on scheduling problems 58
5.1.4 Objective functions in scheduling 61
5.2 Scheduling Techniques 62
5.3 Artificial intelligence 66
5.2.1 Agents 66
5.2.2 Artificial intelligence, machine learning, and deep learning gents 67
5.2.3 Machine learning 69
5.2.4 Reinforcement learning 74
5.2.5 Actor-critic in reinforcement learning 80

Chapter 6 - Application Development 87

6.1 Construction of the environment 87
6.1.1 The step method 87
6.1.2 The reset method 89
6.2 Architectural consideration for deep learning 89

6.2.1 Activation functions 89
6.2.2 Loss function 91
6.2.3 Optimizer 91
6.3 Convolutional neural networks 92
6.4 Architecture for actor-critic implementation 93
6.5 Contribution architecture 96
6.6 Production system definition 96
6.7 Criteria for scheduling technique 99
6.8 Actor-critic approach in Python 99
6.8.1 Technical implementation 100
6.8.2 Functions and parameters definition 102
6.9 Experiments 110
6.10 Results 111
6.11 Analysis of results and discussion 112
6.11.1 Technical perspective analysis 114
6.11.2 Managerial perspective analysis 120
Chapter 7 – Conclusions, limitations, and future work **128**
Bibliography **132**
Annexes **152**
Annex A – Contribution developed code 152

List of Abbreviations

A2C	Actor-Critic algorithm
ADAM	Adaptative moment estimation
AI	Artificial intelligence
ANN	Artificial neural network
CA	Clustering analysis
CNN	Convolutional neural network
CO	Combinatorial optimization
CVRP	Capacitated vehicle routing problem
DJSSP	Dynamic job shop scheduling problem
DL	Deep learning
DMOFWA	Discrete multiple-objective fireworks algorithm
DQN	Deep Q Network
DRL	Deep reinforcement learning
ECT	Earliest creation time
EDD	Earliest due date
ERP	Enterprise resource planning
FCEA	Fuzzy correlation entropy analysis
FCFS	First come first serve
FIFO	First in first out
FL	Fuzzy logic

GA	Genetic algorithm
GEP	Gene expression programming
GP	Gaussian processes
LCT	Longest creation time
LIFO	Last in first out
LPS	Longest process sequence
LPT	Longest processing time
LTPT	Longest total processing time
LTWR	Least total work remaining
LWT	Longest waiting time
MILP	Mixed-integer linear programming
ML	Machine learning
MOD	Modified operation due date
MOFSP-SDST	Multiobjective flow shop scheduling problem with sequence-dependent setup time
mTSP	Multiple traveling salesmen problem
MTWR	Most total work remaining
NCO	Neural combinatorial optimization
NN	Neural network
OBL	Opposition-based learning
PPC	Production, planning, and its control

PS	Process sequence
PT	Processing time
ReLU	Rectifier linear unit
RL	Reinforcement learning
RNN	Recurrent neural network
SLR	Systematic literature review
SPS	Shortest process sequence
SPT	Shortest processing time
STPT	Shortest total processing time
SWT	Shortest waiting time
TD	Temporal difference
TPT	Total processing time
2PTPlusWINQPlusNPT	2Processing Time + Work in Next Queue + Next Processing Time

List of Tables

Table 1. Keywords constructs for database search. 15
Table 2. List of exclusion criteria. 17
Table 3. Multidimensional evaluation for the SLR.....23
Table 4. Flow shop and job shop characteristics.....57
Table 5. Dispatching rules classification. 63
Table 6. Parameters for running the experiment.....111
Table 7. Processing time matrix.....111
Table 8. Result comparison between experiment 1 and SPT rule.....117

List of Figures

Figure 1. Research design structure 5
Figure 2. SLR scheme. 9
Figure 3. Analytical Framework. 10
Figure 4. PRISMA method steps. 16
Figure 5. Distribution of articles per year and per source.....17
Figure 6. Distribution of articles when applying exclusion criteria (1). 18
Figure 7. Distribution of articles when applying exclusion criteria (2). 18
Figure 8. System features main results. 20
Figure 9. Implementation feature results. 21
Figure 10. Performance parameters results. 22
Figure 11. Term's co-occurrence map. 26
Figure 12. Term's co-occurrence time-line map. 27
Figure 13. Keyword's co-occurrence map. 28
Figure 14. Time trend for the construction related to the AI approaches. 29
Figure 15. Time trend keywords technicity. 30
Figure 16. Time trend keywords applicability. 30
Figure 17. Productivity per country. 31
Figure 18. Cites per country.....32
Figure 19. Co-authorship network. 33
Figure 20. Flow shop diagram for three stages in series.....54
Figure 21. Job shop diagram with five work centers and three jobs with different routes. 56
Figure 22. a) Software agent. b) Hardware agent (adapted from Ertel, 2017). 67
Figure 23. Categories inside AI (adapted from Hurwitz & Kirsch, 2018). 68
Figure 24. Basic NN (adapted from IBM Cloud Education, 2020). 70
Figure 25. Design for a RL system (adapted from Sewak, 2019).....74
Figure 26. Conceptual design of the actor-critic method (adapted from Sewak, 2019). 84
Figure 27. Actor-critic implementation 94
Figure 28. Flow diagram of the architecture for the solution developed. 97
Figure 29. Flow diagram for Step function. 105

Figure 30. Flow diagram for Done function. 106
Figure 31. Flow diagram for reward computation. 108
Figure 32. Simulation results. 1) Total reward. 2) Machine utilization. 3) Flowtime. 4) Reward flowtime. 113
Figure 33. Results benchmarking. Top: SPT. Bottom: RL method. 119
Figure 34. SPT rule vs contribution developed performance comparison.....120
Figure 35. Performance dimension analysis impact (flowtime and jobs done). 123
Figure 36. Performance dimension analysis impact (machine utilization). 125
Figure 37. Factors affecting the cost-accuracy trade-off. 126

Executive Summary

1. Introduction

There are some value-adding activities in the manufacturing process as Production Planning and Control (PPC), which can be improved. Due to its manual nature, human interaction has been fundamental in decision-making in those kinds of activities. One of the biggest challenges among the PPC functions is the scheduling problem. This is because the leveraging of resources and the minimization of huge inventories can impact millions of dollars per day.

In this manner, artificial intelligence (AI) is one of the latest approaches facing the scheduling problem. This approach offers automation of different activities in a production plant, which reduced human dependency by machines, software, etc. Prediction, classification, and decision-making are examples of the different functions that an AI approach could perform.

Bearing the exposed ideas in mind, it is possible to state that an advanced method could solve the production scheduling problem, such as AI approaches. Moreover, those methods provide new advantages highly recommended in complex production systems due to the high availability of data in this Industry 4.0 context.

This thesis proposed applying one of the AI approaches by developing a tool that can solve the scheduling problem for a complex production system. In this way, the tool was built considering the different objectives that a company could set. In this case, the

principal purpose was to minimize the flowtime and to maximize the machine utilization of the analyzed manufacturing system.

2. Research Design

To organize this thesis, we designed a research plan based on two phases: i) research objectives for the systematic literature review (SLR) and ii) research methodology. In the first phase, we set three main objectives to do the SLR:

- Understanding AI and production scheduling
- The advantages of their integration
- The improvements of AI over manufacturing systems

To accomplish these objectives, we proposed an analytical framework that contains all these research questions. In the end, we could present a qualitative and quantitative assessment of the results to identify possible solutions. Regarding research methodology, we proposed the following structure: literature review gap examination, the definition of thesis objective, theoretical background related to production systems, scheduling techniques and AI approaches, developing of thesis contribution, discussion of results, and conclusions, limitations, and further research. This planning was the guide followed to do good research and obtain the results we were aiming at.

3. Systematic Literature Review

The SLR was conducted to find trends and gaps in the integration between AI approaches and scheduling problem. Using as a guideline the SLR done by Bueno et al. (2020), we divided this analysis in three stages:

1. **Planning the review:** In this stage, we used the analytical framework from the methodology to find a common point between AI approaches and the scheduling problem, identifying the advantages of AI in production scheduling and finding existing gaps. In this manner, the research questions proposed were:

    - What are the advantages for production scheduling that can be provided by AI?
    - Does the integration of production scheduling with AI determine performance improvements?

    Then, we set different dimensions to assess the relationship between these two inputs by measuring performance improvements such as cost, flexibility, profitability, complexity, etc. Later, regarding the system features, we classified them in system type, applicability, and implementation features, to explore in which sectors the AI approaches, and production scheduling integration was explored or not.

2. **Conducting the review:** With a construction of keywords related to our research, we extracted information from 4 databases: Scopus, Science Direct, Web of Science, and IEEE. After obtaining this information, we applied a filter using a list of inclusion and exclusion criteria (PRISMA method). We saved the information of 54 articles that passed all the filters, and we analyzed it in the next stage.
3. **Reporting and dissemination:** In this stage, the analysis of system features results highlighted two main points. First, 81% of articles applied genetic algorithm (GA) to solve scheduling problems. Second, it was possible to see a significant transfer of knowledge from the academy to the industry even when the former develops methods more advanced to those that the industry can adopt. Regarding the implementation features, the scheduling problem was usually solved by coding for systems made of several machines. Besides, the performance indicator the most representative in applying AI in production scheduling problem is the lead time due to its direct impact on the minimization of the makespan in a production system.

After this analysis, we answered the two research questions. First, concerning the advantages of production scheduling that AI can provide, it is possible to structure an appropriate, tailored, and systematic method for production scheduling to achieve any scheduling objectives. In this way, the AI provided versatility in terms of different solutions. This characteristic is an advantage over traditional techniques because AI approaches allow getting close to real-world manufacturing scenarios. Second, concerning

the possible performance improvement that arises from the integration of production scheduling with AI determined, we found that in many cases, we can achieve one or more scheduling objectives simultaneously, doing a trade-off between them such as makespan, costs, lead time, etc.

Entering the qualitative assessment from the bibliometric analysis, we analyzed the meta-data generated in the SLR. Then, it was possible to conclude that the terms more used in the articles we studied were system, product, control. Furthermore, about keywords, the most frequent strings were “production control” and “genetic algorithm,” and analyzing the keywords construction, we can see that machine learning (ML) is becoming a trend in the latest years. Finally, the productivity per country related to the production of articles on each of them is linked to highly industrialized countries such as China, USA, Iran, and Turkey.

4. Systematic Literature Review Result Discussion

We obtained trends and gaps from the SLR results and the meta-data analysis to solve a scheduling problem using AI methods. The trends that we highlighted are the attractiveness of scheduling problem for multiple machine systems in real-world industrial applications and the reduction of makespan. Concerning the gaps, we found a lack of flexibility in the solutions developed so far compared to the scheduling process in a real-world situation. Besides, we found that solutions through machine learning (ML) techniques have not been explored enough.

Considering these two points, we could adopt a different approach for solving the scheduling problem, recognizing that scheduling suffers changes in its function due to today's market demand. On the one hand, from the bibliometric analysis, we stated that ML was the most promising method to solve the scheduling problem in a changing manufacturing environment. Besides, ML methods did not have constraints to apply them in a production system. On the other hand, we found a gap in flexibility that should be fulfilled given the needs of the current manufacturing conditions (more variety, small batches, customization, etc.), increasing the complexity of the production systems. Moreover, from the performance analysis, we determined that the main objective of the solutions proposed in the reviewed articles was the reduction of makespan. Therefore, we could conclude that the integration between production scheduling and AI was beneficial, and there was a chance to cover the gap of flexibility innovatively through ML methods.

We divided the literature in two by doing new research with production scheduling and machine learning integration as constructs. The first part related to ML approach as an immediate solution to the scheduling problem. And the second related to ML approach as an indirect solution for the scheduling problem. Focusing on the ML approach as a direct solution, reinforcement learning (RL) methods were used when the set problem involved flexibility to face dynamic behaviors in the production systems, reducing the computational cost. On the other hand, for the ML approach as an indirect solution, the most frequent approximation was supervised ML approaches to predict events that triggered changes in an initial schedule.

Then, we set the motivation and objectives for this thesis. So, we selected RL as the most suitable method for addressing the flexibility needed for scheduling, considering the need for simulation time reduction from the SLR results. Furthermore, from a managerial perspective, the scheduling problem is a cost determinant for a company since it defines the excellent assignment of resources and their utilization, translating this into saving and gaining competitive advantage. On the other hand, from the technical perspective, the usage of AI for scheduling improves the production scheduling process due to less human intervention in the decision-making process. Also, AI deals with industry changes managing them through forecasting, data analysis, and simulation. For these reasons, AI could be the enabler for solving the scheduling problem in a dynamic environment. Based on the motivation, the general objective of this thesis was:

"Development of scheduling tool based on AI approach using (static) process-time based scheduling techniques and actor-critic as RL method applied on a parallel machines production system with intermittent job order arrivals."

To achieve this objective, we stablished specific objectives considering three axes: i) the development of a tool, ii) consideration of simulation time, and iii) comparison of results with other methods reported in the literature. Regarding the tool, it was necessary an instrument to run experiments and obtain results; a tool for manufacturing systems with similar characteristics previously seen in the SLR. Furthermore, concerning the simulation

time, this thesis provided a solution to the scheduling problem. Currently, there are exact methods to solve the scheduling problem for the production systems that we analyzed. So, the computational cost resulted as a parameter to consider. Finally, we made a comparison of the results to measure the feasibility of the proposed solution. In this way, we compared performance improvements, simulation time, and managerial performance.

5. Theoretical Background

To develop the tool, it was necessary first to understand the scheduling problem, the production systems in which it is possible to apply the tool, its constraints, and the different scheduling techniques that already exist.

For production scheduling, there are different production systems which different objectives and constraints. In this thesis, we focused on parallel machines with different speeds. So, there are m machines capable of processing jobs with varying times of processing. Because of this characteristic, it is necessary to perform good scheduling to have the rightest sequence to complete different assignments with the available resources in a limited time frame. The constraints in this production system are related to preemptions, job families, or machine eligibility restrictions. Usually, the objective function to be minimized is related to the total job completion time, hence their scheduling. The most usual objective functions are the makespan, the maximum lateness, and the total weighted completion time.

For scheduling techniques, they are classified by exact and approximate methods. The exact techniques have the objective to find the optimal solution. However, their efficiency is limited on NP-Hard problems. On the other hand, the approximate methods take less time to find a reasonable and feasible solution in large scenarios. However, the solution could not be optimal. In the approximate methods, there is a list of dispatching rules that could be static or dynamic. Some of the most used are Shortest Processing Time (SPT) and First In First Out (FIFO). The former is used to minimize the work in progress (WIP) and increase the throughput, starting from the jobs with minimum processing time on a specific machine. The later, used to attend jobs depending on their arrival time.

About AI, one of the methods that resulted most useful was RL. This system has a learning-action loop in which an agent interacts with its environment to select the best possible action given a determined state. First, the action chosen by the agent changes the environment's state and generates a reward for the agent. Later, the agent selects the best possible action for the new state and obtains a reward. In this loop process, the agent tries to improve its decisions when choosing the best action it can take in each environment's state, looking at its rewards in the training process. In the context of RL, there is an Actor-Critic method whose main objective is to find the best policy possible. Here, the critic evaluates the current policy prescribed by the actor, and the actor approximates and updates the value function using samples. Then, the value function is used to update the actor's policy parameters to improve the performance.

6. Application Development

To create the tool, we constructed an environment whose purpose is to present the state and provide relevant responses to train the agent. It has two primary functions called "step" and "reset". In this step function, the input is an action, and it returns a set of elements: state, reward, and done. The reset does not have arguments and return variables and a state corresponding to the start of a new episode.

Regarding the architecture considered for deep learning, we used activation functions such as Softmax, Linear, and Rectifier linear unit (ReLU), loss functions. Besides, we used Adam optimizer that is efficient, fast, and robust against stagnation in a local minimum, gaps, and saddles. We also used convolutional neural networks based on the mathematic operation. To implement the architecture for actor-critic, it was necessary to define an action and policy. In this method, the actor requires a policy that is used to make an action in every step of the training process. At the same time, the actor improves and updates its policy through the policy-gradient approach. The policy is updated continuously using the state value estimation received from the critic.

Concerning this thesis, the architecture of the developed model was based on the actor-critic approach. We defined a parallel machine production system with intermittent order job arrivals with 6 machines, 12 job clusters, and a list of 100 jobs from a pool of 500 jobs to process. The objective function is to minimize the makespan by minimizing the flow time and maximizing machine utilization. We chose a hierarchy of dispatching rules process-time based: SPT and FIFO to achieve this, respectively.

To develop the tool, we used Python using several libraries such as *pandas*, *numpy,* and *torch*. The elements of the environment were defined as follow:

- **State:** vector composed of the jobs' status and the job distribution time.
- **Action:** binary vector (1 for an available machine, 0 for an unavailable machine).
- **Step:** from which the agent will learn how to obtain the highest possible reward given a certain initial state, making a certain action. In this function, we establish several parameters.
- **Done:** it is a Boolean expression which finalize the experiments or episodes when a condition is achieved. In this thesis, the condition is if the number of left jobs pending to do is equal to zero, done becomes True.
- **Reward:** the reward is expressed by a linear combination of two indicators: the flowtime and the machine utilization. The flowtime needs to be normalized through a logarithmic function (0, 1) to be comparable with the machine utilization.

Having all the code developed, we run experiments, and we obtained results that were analyzed from technical and managerial perspectives. The technical view aimed to discuss the profiles over episodes for total reward, machine utilization, flowtime, and reward-related to flowtime, considering the logic that best describes the problem's solving process. Besides, we discussed the quantitative improvements obtained when solving the problem (scheduling) through the tool developed compared to SPT rule. Complementary,

the managerial and business perspective considers the trade-off between the technical results and the advantages/disadvantages of integrating an AI technique to solve the scheduling problem for the business objectives.

In the developed model, we used as reward function the following expression:

$$\boldsymbol{Total\ reward} = 0.5\ (reward\ machine\ utilization) + 0.5\ (reward\ flowtime)$$

The technical perspective analysis was developed by analyzing two points. First, we analyzed the reward function based on the obtained results that better explain the agent's behavior when solving the scheduling problem. Second, benchmarking the results for machine utilization and flowtime using the results obtained in this thesis and those when the problem is solved through SPT dispatching rule.

For the reward function analysis, it was possible to observe that the total reward increased after an unstable initial learning phase. Following episode 2000, it showed a steady-state behavior from episode to episode. This experiment presented the highest difference between the total reward for the transient and steady states. For the machine utilization, this indicator increased over time (expected behavior), and the obtained profile contained defined transient and steady states after the episode 2000. For the resulting mean flowtime profile, we observed a decreasing profile over episodes. Therefore, it is possible to state that the agent is learning. Besides, it was possible to observe a defined transient and steady-state for the values obtained over time. Lastly, for the reward associated with

the flowtime, the obtained profile showed an increasing behavior, where it was possible to identify a steady state after the episode 2000. In conclusion, this experiment's results are consistent in all the measured dimensions and fulfill the requirements stated in the main objective of this thesis.

For the performance analysis, we benchmarked the obtained results for machine utilization and flowtime. The comparison made used the results obtained from solving the same problem using SPT rule. From the results, we highlighted that:

- There was a slight improvement in the average machine utilization (6.1%). When the scheduling problem is solved through SPT rule, the machine utilization resulted in 91.8%. When the problem is solved using the solution proposed in this thesis, it resulted in 97.4%. However, the standard deviation decreased 97.8%, which reduced the coefficient of variation by 97.9%.
- The average flowtime presented a reduction of 53.9%, with a decrease in the coefficient of variation of 77.5%. The flowtime reduction was expected since the contribution developed was designed with a reward function that minimizes this performance indicator.
- Compared to the solution provided by SPT rule, the contribution developed in this thesis presented an increment in the simulation time of 692.1%. This is because RL techniques require a large number of samples for the learning process.

Due to the resulting changes in performances, an analysis from a managerial perspective was necessary. In this way, we analyze them from a manufacturing operations management perspective and their relationship with the business strategy and impact on business objectives. The dimensions that we chose to do the analysis were: i) time, ii) price (cost), iii) quality, iv) flexibility, and v) service. To make this analysis clearer, we used as an example the aerospace industry since we found in the literature the available business strategies that companies have followed to satisfy the market needs. This analysis permitted us to conclude that:

- For those organizations that based their strategy on bespoken development or highly tailored product line (significant investments, high quality, small numbers of units), the minimization of the flowtime impacts positively performances related to time (delivery speed) and cost (inventory carrying cost, revenue opportunity cost, and variable tooling cost) (Chao & Graves, 1998). In this way, the business gains competitive advantage. Besides, it is possible that the increment of the machine utilization positively impacts performances related to cost (reduction of fixed costs). However, since these strategies do not rely on quantity produced, increasing the utilization could not represent an increment in competitive advantage.
- For those organizations that based their strategy on at-rate product line or global-scale and scope (low cost, delivery schedule stability, high production volume), the minimization of the flowtime impacts positively (in a higher level) the number of produced units. This fact could improve performances related to cost (benefits from

economies of scale) and service (higher availability of goods) (Slack & Brandon-Jones, 2018). So, the business could gain competitive advantage. Furthermore, the increment in the machine utilization has a higher positive impact on performances related to cost (reduction of fixed costs) and service (availability of units produced) since these strategies are quantity dependent. So, the business has an enormous potential to gain competitive advantage.

Regarding the simulation time, since the contribution proposed in this thesis is based on a RL technique, the computational cost increments naturally. However, computer cost optimization based on the fundamentals of the algorithm is possible, but it is out of the scope of this thesis.

7. Conclusions, limitations, and future work

In this thesis, a ML technique was applied to develop a tool to solve the scheduling problem for a specific manufacturing system. First, we did research based on a SLR. As a result, we found that integrating AI approaches and production scheduling generates different advantages in terms of less human intervention, effective methods, and the solution for the manufacturing system when solving the scheduling problem. Moreover, this integration allowed performance improvements in cost, productivity, lead time, and many other indicators. In this manner, the SLR opened the doors to new ways to solve the scheduling problem by analyzing gaps and trends in which the objective function for the

problem addressed was the minimization of makespan. After a deep analysis, we chose the most suitable AI method, and we set the objectives. Then, we made a thorough revision of ML techniques in the literature to solve the scheduling problem. In this revision, we found that RL was used for manufacturing systems that face changes and need flexibility while reducing the computational cost.

Considering the previous results, we constructed a solution in Python based on RL method that solves the scheduling problem for a parallel machine system that processes jobs with intermittent arrivals. Then, we compared the solution that provides SPT, and the algorithm developed in this thesis through two indicators: machine utilization and flowtime.

From the technical perspective, we concluded that the method presented in this thesis resulted in s a slight improvement for the average machine utilization (6.1%), with a 97.8% reduction of the standard deviation and 97.9% for the coefficient of variation. Moreover, the average flowtime presented a reduction of 53.9%, with a decrease in the coefficient of variation of 77.5%. Finally, the contribution developed presented an increment in the simulation time due to the large number of samples required for the learning process in the RL techniques.

From the managerial perspective, we concluded that the obtained results must be overlapped with the manufacturing system business objectives to determine if they are beneficial or harmful. For example, if an organization based its business objectives on satisfying market needs related to unique products with high quality, the minimization of

the flowtime positively impacts time (delivery speed) and cost (inventory carrying cost, revenue opportunity cost, and variable tooling cost). Otherwise, if an organization based its business objectives on satisfying market needs related to massive production where the cost is a competitive advantage, the minimization of the flowtime has a higher positive impact on the number of units produced. Furthermore, for the resulted increment in machine utilization, we considered the changes that impact costs. So, we concluded that for an organization that based its business objectives on satisfying market needs related to unique products with high quality, the increment of the machine utilization impacts positively performances related to fixed cost. Contrary, for organizations that based their business objectives on satisfying market needs related to massive production, machine utilization positively impacts the cost (reducing the fixed cost) and service (availability of units produced). In conclusion, it was possible to state that there are benefits from integrating an AI-based method with a traditional company function. However, they must be analyzed technically (where changes in performances are seen) and from a managerial perspective (where the impact on business objectives is observed).

As limitations, we considered two points that could not permit using the proposed solution in this thesis in a real-world case. The first point is that we only considered one among many events in a real-world manufacturing scenario (intermittent job order arrivals). So, the algorithm was trained to face only this type of change successfully. The second point is related to the increment of the simulation time this type of algorithm requires. Although the contribution developed faces rapid changes related to the arrival of jobs, the training

needs a lot of time. If we considered more changing factors (maintenance, quality issues, cancelation of order, etc.), the simulation time (including the training) would be higher than the result obtained in this thesis. Therefore, for future work, applying AI approaches to solve scheduling problems must be explored more and more to improve technical performance and fulfill managerial terms (i.e., solving the scheduling problem for job shops).

Chapter 1 - Introduction

In a manufacturing company, high amounts of raw material (inputs) are turned into many products (outputs), and this process represents millions of dollars per day. For this reason, it is essential to define how much, when, and what to produce to satisfy the client's demand (Hubbs, Sahinidis, Grossmann & Wassick, 2020). Production planning and its control (PPC) is the activity that answers these questions (Oluyisola, Sgarbossa & Strandhagen, 2020). However, this activity needs to be continually updated to the production environment's changes, such as the market's requirements (quality and performance), the introduction of new products, production asset breakdowns, lack of demand, etc. (Bueno, Godinho Filho & Frank, 2020). So, to face the continuous update requirement, in the last years the PPC function has been supported by different tools, such as information and communication technologies; these tools uphold demand forecasting, operations planning and sales, production scheduling, etc.

Production scheduling, as part of the PPC function, is the decision-making process that aims to find a way to assign and sequence tasks that utilize resources (e.g., labor, material, and equipment) considering production constraints and the minimization of costs. In industry, the scheduling process is mainly executed by software packages assisted by experience human schedulers that implement distinct scheduling philosophies (such as manufacturing resource planning, just-in-time, or optimized production timetables) (Rovithakis, Perraki & Christodoulou, 2001; Akyol & Bayhan, 2007). Scheduling is a complex task that aims to optimize activities reaching one objective (or multiple objectives)

through available production data, including historical from previous planning. It is affected by the production planning system, which considers medium to long-term planning for the company (Pinedo, 2016). Hence, decisions at the planning level affect the scheduling process. From a mathematical point of view, scheduling problems are classified as NP-hard problems, for which many solving algorithms exist, including artificial intelligence approaches (Parente, Figueira, Amorim & Marques, 2020).

Artificial intelligence (AI), as one of the approaches for solving the scheduling problem, refers to a human-like intelligence performed by a computer, robot, or another type of machine. It is popularly defined as a computer's ability (or machine) to imitate the human mind's capabilities. It is related to, for instance, learning from experiences, recognizing objects, understanding and responding to language, making decisions, solving problems, among others (IBM Cloud Education, 2020). The definition of AI has evolved over the years based on the industries' focus of utilizing this technology. Simultaneously, large amounts of data and the analogous development and availability of computational capacity had enabled the development of AI technologies, which allows a faster and more accurate process capacity. AI is one of the enabling technologies of industry 4.0 and, for the future workforce, machines are meant to take over some jobs. However, as occurred in the previous industrial revolutions, this new technology will bring new jobs as well, some of which are yet to be realized; and it should be seen as an instrument to help humans do things faster and more efficiently (Siemens, 2021).

Today, in the context of industry 4.0, there is a threat that radically changes the traditional PPC (that includes the scheduling problem) paradigm. Industry 4.0 traditional view considers the manufacturing process as the core activity, so the PPC function also faces changes due to industry 4.0 technologies (Bueno et al., 2020). As a managerial function, PPC could be transformed into a more integrated and automated process by using digital technologies. Some traditional activities of the PPC function could be affected by, for example, the integration between the physical and digital production environments (cyber-physical systems and digital twins), the integration of the enterprise resource planning (ERP), manufacturing execution system, artificial intelligence, and machine-to-machine approaches for ordering systems (Bueno et al., 2020).

This thesis was developed through a systematic literature review (SLR) considering the improvement performances for a manufacturing system when the scheduling problem is solved through an AI approach. In this way, all the literature findings are presented. Later, an AI-based application was developed to solve the scheduling problem in a system to validate the SLR findings and contribute to the knowledge needed to face the change of paradigm that PPC function meets in the industry 4.0 context.

The structure of this thesis is the following. Chapter 1 presents the introduction of this thesis. Chapter 2 describes the research design that set the justification for performing a SLR, and then it shows the research methodology followed in this thesis. Chapter 3 presents the SLR method, its development, and the obtained results. Chapter 4 reports the critical discussion of the gaps and trends found in the SLR and the analysis of the literature's

available information related to them. Besides, it presents the definition of the objectives for this thesis. Chapter 5 covers the theoretical background needed for the development of the proposed AI-based application that solves the scheduling problem. Chapter 6 comprises the description of the developed application, its validation, and the results when using it. Lastly, chapter 7 presents the conclusions, limitations, and future work.

Chapter 2 - Research Design

This chapter presents the structure for planning this research thesis into two phases: i) the research objectives for the SLR, and ii) the research methodology.

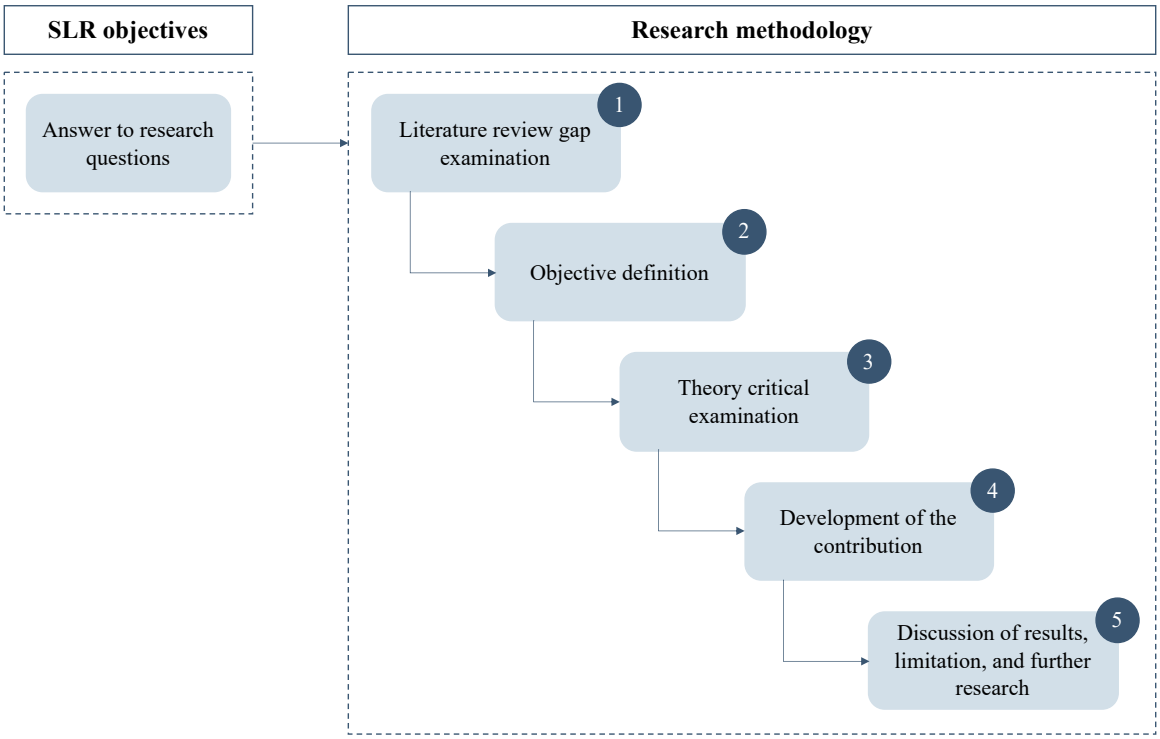

Figure 1. Research design structure

2.1 Systematic literature review research objectives

For this thesis, we performed the SLR aiming at: i) a better conceptual understanding of AI and production scheduling, ii) the advantages related to this integration, and iii) performance improvement of AI over manufacturing systems. In this

manner, it was required to set an analytical framework based on research questions that allowed to achieve the SLR objectives. After the development of the SLR methodology, it delivered content and bibliometric analysis through qualitative (graphs) and quantitative (tables) representations related to the different dimension of the SLR that answer the research questions satisfactorily. Moreover, following the qualitative and quantitative assessment of the results, it was possible to identify fields not explored enough allowing us to go further in the research methodology phase.

2.2 Research methodology

After result obtained from the SLR, we designed the following structure: i) literature review gap examination, ii) definition of the thesis objective, iii) critical examination of theory related to AI approaches, production systems, and scheduling techniques, iv) development of the contribution, v) discussion of the results, and vi) conclusions, limitations, and further research.

2.2.1 Literature review gap examination

In this part, the gaps found in the preliminary stage are explored in deep. To do so, we developed a new and general research on the databases using a new keyword construction related to the gaps. After this, the obtained information was retrieved and discussed to characterize the problem studied by each author.

2.2.2 Objective definition

In this section, we proposed an objective that pursues an alternative solution for the scheduling problem than those reported in the literature. Later, we presented the specific objectives to achieve the main goal of this thesis.

2.2.3 Theory critical examination

Next, we analyzed the theory dividing it into three parts:

- **Production system analysis:** This is the characterization of the different production systems and the comparison between their behavior in terms of scheduling issues.
- **Scheduling techniques:** This is the analysis of existing methods, their classification, and their application for different scenarios.
- **Method analysis:** This is the understanding and application of AI methods and their feasibility for solving the scheduling problem.

2.2.4 Development of the contribution

We developed a solution in Python considering the characterization of the problem made in the previous steps. We chose Python since it is a user-friendly programming language and open-source tool useful in AI applications.

To set the conditions to develop the contribution, we proceeded to select the production system, following as criteria its relevance in terms of performance and impact for the future of manufacturing. We also considered the high criticality of the scheduling

problem in the selected production system. For choosing the most appropriate scheduling technique, we considered the achievement of the objectives proposed in this thesis. Moreover, the approach selection was based on the method's performance in the scheduling process's decision-making and the amount of data required to have reliable results. Another motivation to select a determined approach was given by the gaps found in the SLR.

2.2.5 Discussion of the results, limitations, and further research

To obtain results, we experimented with a comparable set of manufacturing company data. The result obtained from this experimentation was the production schedule provided by an AI approach by optimizing the chosen objective function and tuning parameters (i.e., computational time, hyperparameters, etc.). Then, we proceed to discuss the tool's performance in different scenarios when achieving the objective function, changing the problem's initial conditions. Furthermore, we identified the limitations that must be overcome and guide a path for future research.

Chapter 3 - Systematic Literature Review Methodology

This chapter presents the SLR we conducted to find trends and gaps in the relationship between scheduling problem and AI approaches for solving it. We related scheduling problems with AI techniques in a framework representing intersections and studies addressed to real-world applications. The SLR used in this project followed the guidelines showed by Bueno, Godinho & Frank (2020), where three stages are defined: i) planning the review, ii) conducting the review, and iii) reporting and dissemination. This chapter presents the development for all the phases.

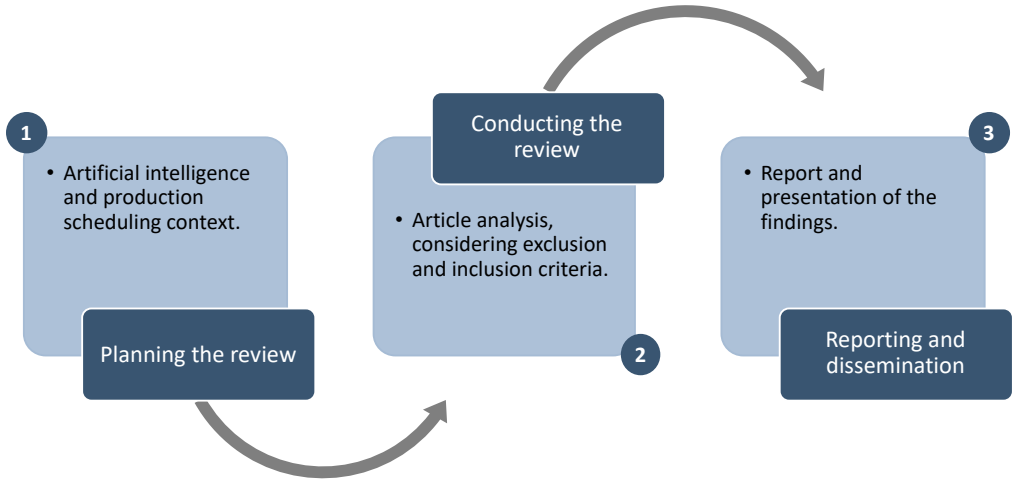

Figure 2. SLR scheme.

3.1 Planning the review

The first step was to determine the nature of AI and its relationship with production scheduling. Then, we identified and defined an analytical framework, keywords, and serach

strings. In this way, for the analytical framework, we set two research questions to create a research protocol for the SLR.

3.1.1 Analytical framework

For this thesis, we performed a preliminary selection of references and research of the literature related to the production scheduling function and AI methods. We evaluated different dimensions related to real-world applications and the performance of the system analyzed. Figure 3 shows how we set the analytical framework.

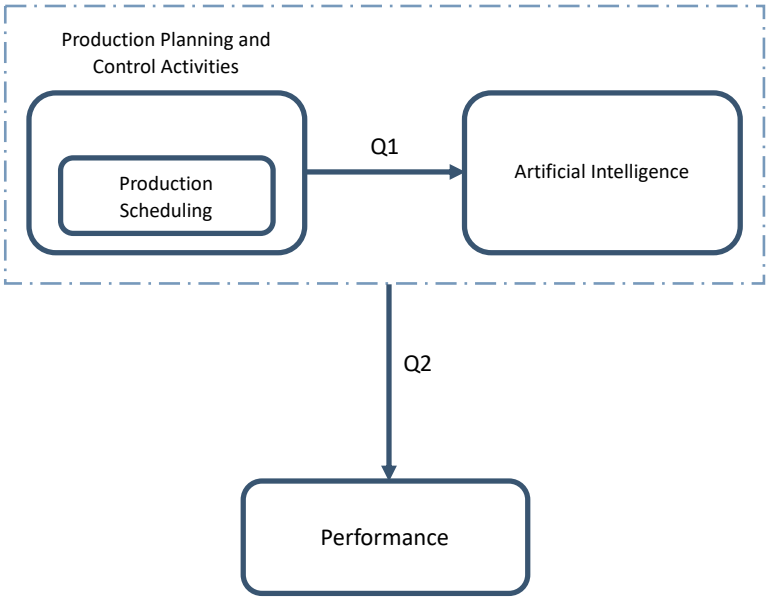

Figure 3. Analytical Framework.

The framework's purpose was to find a common point between the concepts, then identify the contribution of AI into production scheduling, and finally, find the existing gaps. Based on this, it was possible to propose two research questions:

- **RQ1:** What are the advantages for production scheduling that can be provided by AI?
- **RQ2:** Does the integration of production scheduling with AI determine performance improvements?

3.1.2 Performance improvement

We based the selection of performance improvements on the review made by Bueno et al. (2020), where they aggregated different dimensions already identified by other authors. For this project, we used cost, flexibility, productivity, energy and resources, quality, profitability, lead time, robustness, and complexity, to evaluate the relationship between production scheduling and AI approaches.

- **Cost:** We considered a cost improvement if the studied problem related the minimization of a cost function.
- **Flexibility:** If the system studied considered changes in the conditions for solving the scheduling problems, such as machine breakdowns, random incoming work orders, machine maintenance, product quality issues, etc., we considered it as a flexible solution for the studied system.
- **Productivity:** We considered a productivity improvement when the studied problem explicitly related the improvement of an output ratio (number of units, kg of product, etc.) to an input (raw material, labor, and machine usage) over a specific period.

- **Energy and resources:** We considered an improvement in energy and resources when the solution explicitly aimed to save energy and resources.
- **Quality:** If the solution's objective from the scheduling problem improved process/product quality, we included the title into the quality improvement qualifier.
- **Profitability:** We considered a profit improvement if the studied problem related the maximization of a profit function.
- **Lead time:** If the proposed solution for the scheduling problem aimed to minimize the lead time, or improve the makespan, we included the title into the lead time qualifier.
- **Robustness:** Into this qualifier, we considered the studies that proposed a solution that improved the tolerance to perturbations that might affect the system functionally.
- **Complexity:** We considered a complexity improvement when the proposed solution aimed to achieve multiple objectives when solving the scheduling problem. In this way, we thought of complexity reduction when authors explicitly showed multi-objective functions and improvements in computational costs, simulation time, more manageable solution methods.

3.1.3 System dimensions

We defined the selection of system dimensions based on the article made by Cimino, Negri & Fumagalli (2019), where they performed a division for better analysis: i) system features, and ii) implementation features. For the system features, we considered the system type (dynamic/static, discrete/continuous), and the type of application (industry/academy). For the implementation features, we considered industry type, number of machines (single/multiple), improvement orientation (process/product), solution tool (coding/software), and the AI approach used to solve the discussed problem.

System features – system type

- **Dynamic/static:** We considered as a dynamic system the one where its initial conditions continuously change in time. For scheduling, we considered it the system that receives task orders over time steps. Contrarily, we considered the static system as the one where the initial conditions do not change in time.
- **Discrete/continuous:** We considered as discrete those manufacturing systems that produce parts, components, and sub-assemblies. On the other hand, we considered as continuous those manufacturing systems where the materials (i.e., fluids) are perpetually processed, experiencing chemical reactions, mechanical, or heat treatment.

System features - application type

- **Industry/laboratory:** We identified the system as 'industry' when the developed solution answered a problem related to a real-world industry-level problem. Otherwise, we considered the system as 'laboratory' when the developed solution improved a laboratory-scale problem.

Implementation features

- **Industry type:** It defined the industry sector where the AI solution was implemented. We included 'academy' for theoretical or laboratory scale applications.
- **Number of machines:** If the studied system was composed of only one machine, we considered it a single machine system. Otherwise, if the system was composed of more than one machine, we regarded it as a multiple machine system. We did not consider further features for systems composed of more than one resource (parallel, series, redundant, etc.)
- **Improvement orientation:** We considered the type of improvement made when applying an AI approach to the studied problem. If the solution proposed to improve a process variable, we regarded it as an improvement to the process—for example, makespan, throughput time, inventories, idle time, etc. On the other hand, when the solution proposed improved a product variable, we considered it a product improvement—for example, quality specifications of the product.

- **Solution tool:** We looked at the technical way the authors solved the studied problem. If the problem was solved using software, we identified those titles as 'software'. On the other hand, if the authors solved the problem through a coded tool, we identified those titles as 'coding'.
- **AI approach:** We identified the approach inside the AI techniques used by the authors to solve the studied problem.

3.2 Conducting the review

Following the logic for SLR used in Bueno et al. (2020), we based our study on journal articles and conference proceedings. We focused the analysis on information sources that combined production scheduling and AI from the year 2010 onward. The time range was set from January 2010 to October 2020. Regarding databases, we extracted information from Scopus, Science Direct, Web of Science, and IEEE, using the string constructs showed in table 1. We decided to consult on these databases since they host traditional journals of high impact on the research subjects.

Table 1. Keywords constructs for database search.

Artificial intelligence construct	Production scheduling construct	Applied case construct
"machine learning" OR "genetic algorithm" OR "neural network" OR "fuzzy logic" AND	"production scheduling" OR "production planning" AND	"Case study" OR "Industrial application" OR "Industrial environment"

For the construct related to artificial intelligence, we decided to open the concept into 'machine learning' (ML), 'genetic algorithm' (GA), 'neural network' (NN), and 'fuzzy logic' (FL). We considered these approaches since we want to evaluate their applicability in a real-world problem. For the construct related to production scheduling, we used synonyms related to the PPC function. We did the same for the construct related to the application, where we established 'study cases' or 'industry applications'.

After obtaining the search results, we applied the 'Preferred Reporting Items for Systematic Review, and Meta-Analysis' (PRISMA) approach reported by Moher, Liberati, Tetzlaff, Altman & Prisma Group (2009). This approach is comprised of identification, screening, eligibility, and inclusion. Figure 4 shows a scheme of the PRISMA method.

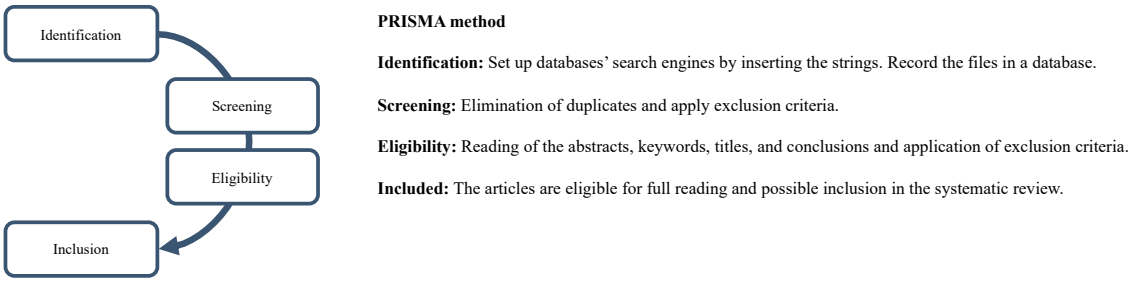

Figure 4. PRISMA method steps.

After obtaining results from the databases, we recorded the information in a spreadsheet. In the identification phase, we obtained 305 documents. Then, we excluded 101 duplicate documents and remained 204 titles. Figure 5 shows the number of articles distribution over time and among the database consulted.

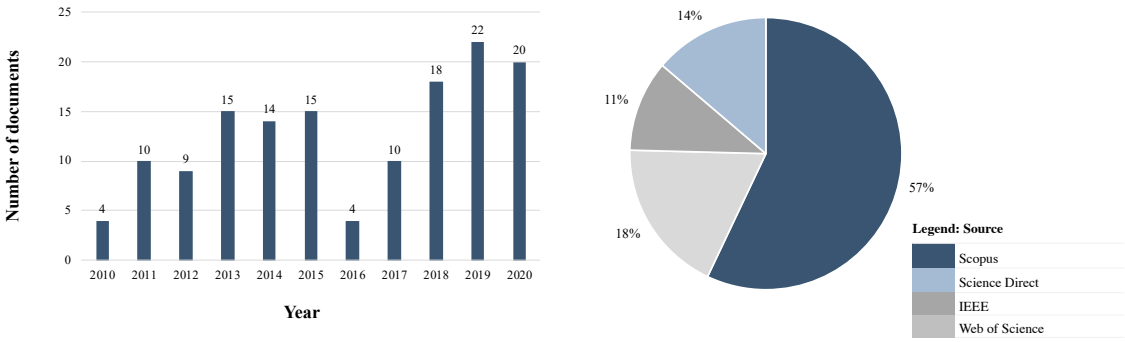

Figure 5. Distribution of articles per year and per source.

Following this process, we applied the exclusion criteria that table 2 shows. The three exclusion criteria (SER, NFT, TRC) were used to filter the 204 identified articles in the screening process. As a result, we excluded 91 titles. Then, we read the abstracts, keywords, titles, and conclusions, for applying the remaining exclusion criteria in table 2 (NR, PR, and CR), resulting in the additional exclusion of 59 articles.

Table 2. List of exclusion criteria.

Criteria type	Criteria	Code	Criteria detail
Exclusion			
	Search engine reasons	SER	SER (1): The paper has a title and abstract in English, but does not have the full text SER (2): The article does not come from an academic journal or conference proceeding.
	No full text	NFT	The paper does not have available text to be assessed
	Time range	TRC	The article was not published within the defined time range: from 2010 to 2020.
	Non-related	NR	The article does not relate the relationships of the Production Scheduling and artificial intelligence.
Inclusion	Partially related	PR	The article generically addresses the subject of Production Scheduling or Artificial Intelligence with an indirect interaction between them.
	Closely related	CR	The research focus of a paper clearly and strictly addresses the integration of Production Scheduling with Artificial Intelligence and its capabilities

In this way, we obtained a final set of 54 papers. These articles were eligible for meta-data bibliometric analysis and complete reading. Figures 6 and 7 show the filtering process graphically when applying the PRISMA method.

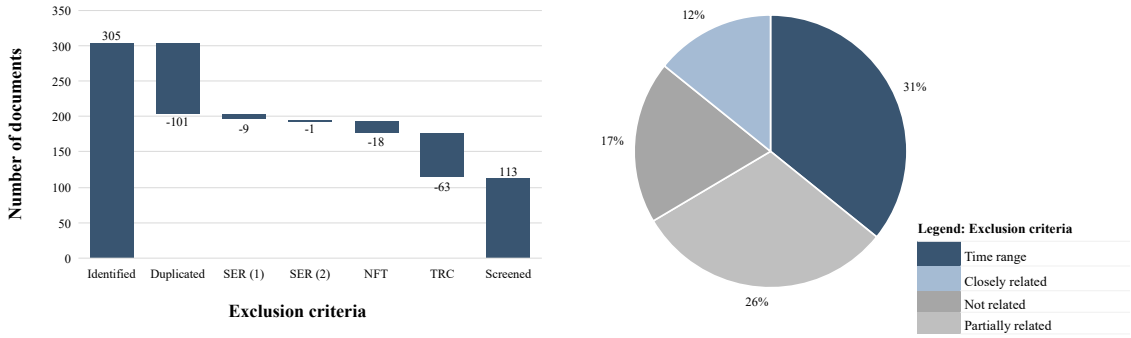

Figure 6. Distribution of articles when applying exclusion criteria (1).

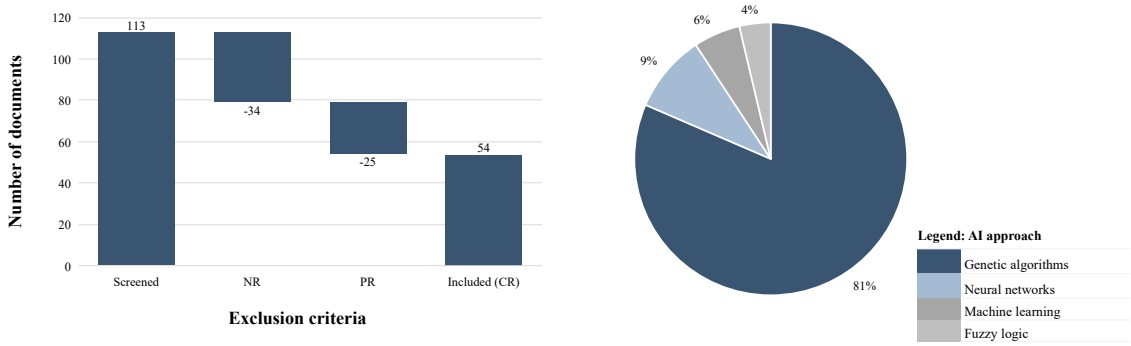

Figure 7. Distribution of articles when applying exclusion criteria (2).

3.3 Reporting and dissemination

This subsection presents the results from the SLR performed for the relationship between scheduling problem and AI approaches.

3.3.1 System features

Regarding the system features, we found a noticeable trend for using GAs when solving the scheduling problem (81% of the total). We found that 50% of the total studies aimed to solve the scheduling problem using GA in an industrial and discrete system. From this percentage, 56% belonged to dynamic systems, and 44% to static systems. Besides, we could observe that at laboratory scale, there is also a visible trend of using the same method for solving the scheduling problem (15%). From these results, we could conclude a significant transfer of knowledge from the academy to industry, even when the academy develops methods more advanced to those that industry can adopt. Furthermore, it was possible to determine that ML, NN, and FL methods have not been explored enough for the period analyzed and in real-world manufacturing environments.

Analyzing the combination of the different dimensions, we could confirm that an industrial-discrete-closed system was the prominent tendency when applying AI techniques for solving the scheduling problem. Regarding industries where AI has been applied, the most significant fields were academy, metal/plastic, food, electronic, construction, and textiles. Figure 8 shows the chart for the analysis results made over the system features and the distribution between industry fields that applied AI approaches.

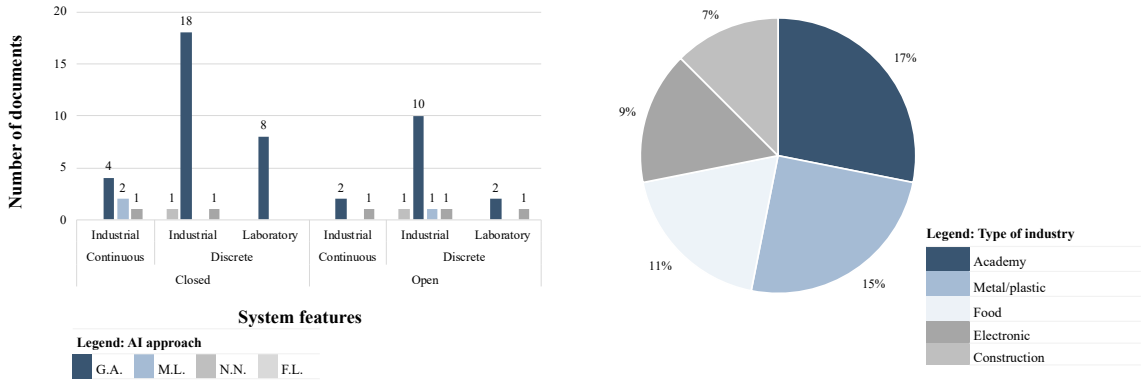

Figure 8. System features main results.

3.3.2 Implementation features

We found that 69% of the studies solved the scheduling problem through coding. In contrast, 31% of the studies used software. Besides, we found a distinct tendency to implement the solution for a process improvement instead of a product improvement (78%). Moreover, we found that 80% of the total titles implemented the solution on several machine systems. In this way, we could conclude that the general trend for the AI implementation as a solution resulted in coding, for the improvement of a process, in a system of several machines (35%). Figure 9 shows the chart for the results of the analysis made over the implementation features.

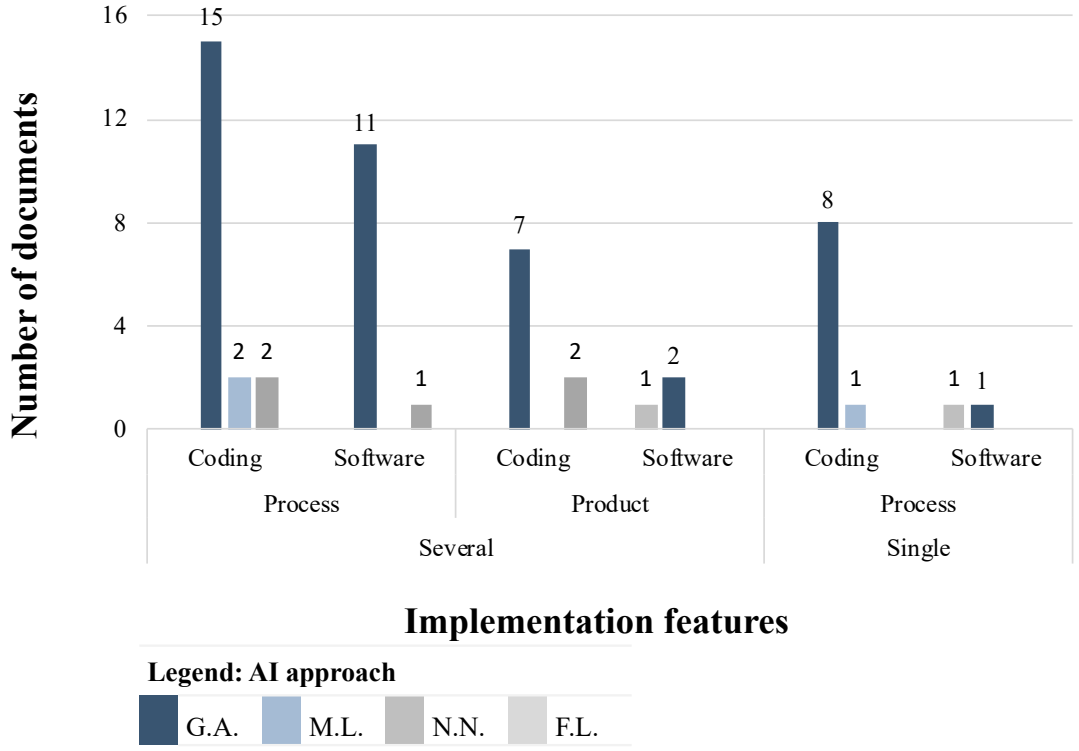

Figure 9. Implementation feature results.

3.3.3 Performance

As stated above, we also analyzed the included articles considering the performance improvements over the studied system. In this way, we found that 26% of the articles aimed at improving the lead time. This is a common trend since one solution to the scheduling problem finds an appropriate makespan under determined constraints (Pinedo, 2016). Therefore, the lead time is improved. Besides, we found that 20% of the included articles worked with a function to reduce cost as objective. Normally, the optimization function costs are those related to times for set-ups, changeovers, and ramp-ups. Moreover, we found that 18% of the titles analyzed aimed at increasing productivity. Once again, this

result is attached to reducing the makespan since it is possible to produce more in the same time frame. Other essential performances found were complexity (9.6%), and energy usage (7.4%). Figure 10 shows the distribution obtained for the performances defined for the SLR.

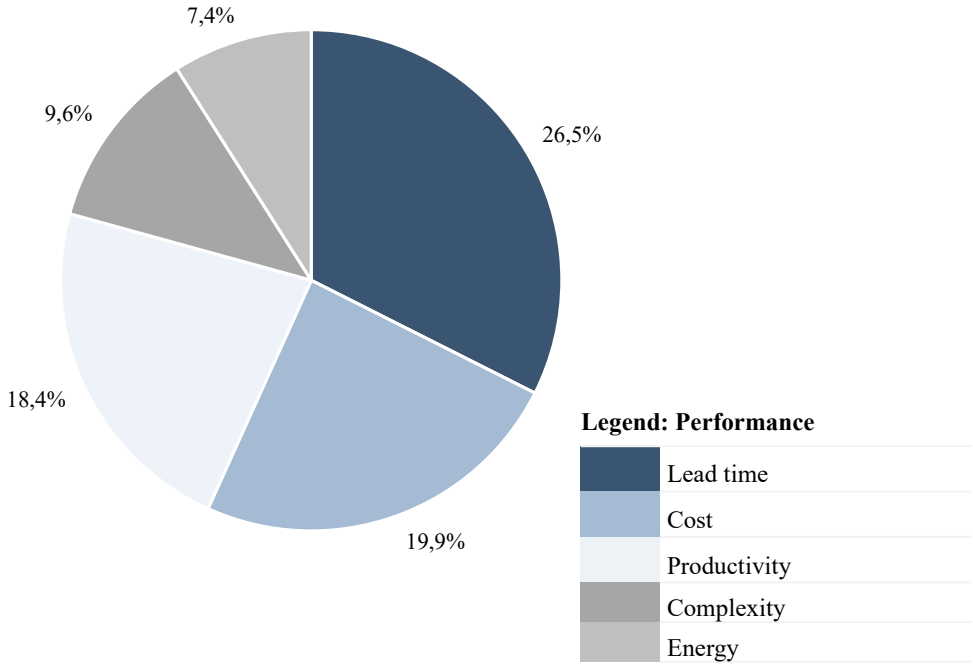

Figure 10. Performance parameters results.

3.4 Answer to research questions

Following, we present the results concerning the two research questions proposed in the analytical framework section. We answered the questions based on the resulting multi-dimension matrix shown in table 3.

Table 3. Multidimensional evaluation for the SLR.

Author	System features						Implementation features											Performance improvement								
					Application type			Number of machines		Improvement		Solution tool		AI approach												
	Dynamic	Static	Discrete	Continuous	Industry	Laboratory	Industry type	Single	Several	Process	Product	Software	Coding	Machine learning	Genetic algorithms	Neural networks	Fuzy logic	Cost	Flexibility	Productivity	Energy	Quality	Profitability	Lead time	Robustness	Complexity
Hubbs, Li, Sahinidis, Grossmann & Wassick (2020)		▼		▼	▼		Chemical	●		●			●	●					◆				◆			
Alipour, Khodaiari, Jafari & Tavakkoli-Moghaddam (2020)		▼	▼		▼		Mining		●	●			●		●			◆					◆			
Ramos, Faria, Vale & Correia (2020)		▼		▼	▼		Energy		●	●			●			●					◆					
Hou, Li, Wang & Hu (2020)	▼			▼	▼		Mining		●	●		●			●				◆	◆				◆	◆	
Lingitz & Sihn (2020)	▼		▼		▼		Semiconductors		●	●			●	●								◆		◆		
Abreu & Prata (2020)		▼	▼		▼		Construction		●	●			●		●					◆				◆		
Bhosale & Pawar (2020)		▼	▼			▼	Academy		●	●			●		●			◆						◆		
Kunapareddy & Allaka (2020)		▼	▼		▼		Equipment		●		●		●		●					◆			◆			
Xu & Song (2020)	▼		▼			▼	Academy		●	●			●		●			◆	◆	◆			◆	◆		
Htwe (2019)		▼		▼	▼		Mining		●	●			●		●			◆					◆			◆
Mangngenre et al. (2019)	▼		▼		▼		Metallurgic		●		●	●					●	◆		◆						
Rubaiee, Cinar & Yildirim (2018)		▼		▼	▼		Energy	●		●			●		●			◆			◆			◆		◆
Jubiz-Diaz, Santander-Mercado & Candelo-Becerra (2019)	▼		▼		▼		Metallurgic		●		●		●		●			◆		◆	◆			◆		◆
Dao, Abhary, Marian & Goh (2019)		▼	▼		▼		Electronic		●		●	●			●			◆		◆						
Sinthamrongruk, Premphet, Smutkupt, Dahal & Smith (2019)		▼	▼		▼		Electronic		●	●		●			●									◆		
Johansen, Bendtsen, R.-Jensen & Mogensen (2019)	▼		▼		▼		Food		●		●		●			●				◆						
Lishchuk, Lund & Ghorbani (2019)		▼		▼	▼		Mining		●	●			●	●						◆						
Shen & Smalov (2018)		▼	▼		▼		Food		●	●			●		●			◆		◆				◆		
Sobottka, Kamhuber, Faezirad & Sihn (2019)		▼	▼		▼		Food		●	●			●	●		●		◆		◆	◆					◆
Yuliastuti, Rizki, Mahmudy & Tama (2019)	▼		▼		▼		Textile		●		●		●		●			◆				◆		◆		
Mirahmadi & Taghipour (2019)		▼		▼	▼		Energy		●	●			●		●			◆			◆			◆		
Wang & Hu (2018)	▼		▼		▼		Electronic		●		●		●		●			◆		◆						
Safari, Hafezalkotob & Khalilzadeh (2018)		▼	▼		▼		Automotive		●	●			●		●			◆						◆		
Liao, Jiang & Zhang (2017)		▼	▼			▼	Academy	●		●			●		●			◆						◆		
Wang, Hu & Gong (2018)	▼		▼		▼		Construction		●	●		●			●			◆						◆		
Rajan & Tjprc (2018)		▼	▼			▼	Academy		●	●			●		●							◆		◆		◆
Wen, Hou, Liu Z & Liu (2017)	▼		▼			▼	Academy		●		●		●		●	●		◆					◆			
Gong X et al. (2017)	▼		▼		▼		Metal/Plastic	●		●			●		●			◆			◆					
Purohit, Kumar, Lad , Manjrekar & Singh (2017)	▼		▼		▼		Automotive		●	●			●		●				◆					◆		◆
Wang & Hu (2017)	▼		▼		▼		Construction		●	●		●			●			◆						◆		
Dao, Abhary & Marian (2017)		▼	▼		▼		Electronic		●		●	●			●			◆		◆						
Zhou, Xu, Cheng, Xu & Jia (2017)	▼			▼	▼		Textile		●	●		●			●						◆		◆			
Liao, Zhang & Jiang (2016)		▼	▼			▼	Academy	●		●			●		●			◆						◆		
Liu, Dong, Lohse & Petrovic (2015)		▼	▼			▼	Academy		●	●		●			●						◆			◆		
Chakrabortty, Hasin, Sarker & Essam (2015)	▼		▼		▼		Textile		●		●		●		●			◆							◆	◆
Worapradya & Thanakijkasem (2015)	▼			▼	▼		Metallurgic		●	●		●				●								◆	◆	
Aydemir & Koruca (2015)		▼		▼	▼		Utilities		●	●		●			●					◆				◆		◆
Castillo & Gazmuri (2015)		▼	▼		▼		Metal/Plastic		●		●		●		●					◆				◆		◆
Costa (2015)		▼	▼		▼		Pharmaceutical		●	●			●		●					◆				◆		
Wang, Tu & Chen (2015)		▼	▼		▼		Metal/Plastic	●		●			●		●			◆		◆		◆		◆		
Hecker, Stanke, Becker & Hitzmann (2014)		▼	▼		▼		Food	●		●		●			●			◆		◆				◆		◆
Liu, Dong, Lohse, Petrovic & Gindy (2014)		▼	▼			▼	Academy		●	●		●			●						◆			◆		
Dai, Tang, Zhang & Yang (2014)		▼	▼		▼		Metal/Plastic		●	●		●			●						◆			◆		
Yao & Pan (2013)		▼	▼		▼		Metal/Plastic		●	●		●			●					◆				◆		◆
Longo (2013)	▼		▼		▼		Metal/Plastic	●		●			●		●					◆				◆		
Cope & Li (2013)	▼		▼		▼		Metal/Plastic		●	●		●			●			◆		◆				◆	◆	
Liu, Chai, Qin, Pan & Yang (2012)		▼	▼		▼		Metal/Plastic		●	●			●		●									◆		
Noroozi, Mokhtari & Abadi (2012)		▼	▼		▼		Electronic	●		●			●		●					◆			◆	◆		◆
Liu, Sun, Yang & Chuang (2011)		▼	▼		▼		Food		●		●		●		●							◆		◆		
Liu, Li, Yang, Wan & Uzsoy (2011)	▼		▼			▼	Semiconductors		●	●			●		●			◆					◆			
Ko & Wang (2011)		▼	▼			▼	Construction		●	●			●		●					◆				◆		◆
Ma & Huang (2018)		▼	▼			▼	Academy		●	●			●		●				◆						◆	◆
Gran & Aitken (2015)		▼	▼		▼		Food	●		●			●		●			◆	◆	◆				◆		
Damayanti, Rachmat & Atmaja (2013)		▼	▼		▼		Aerospace	●		●		●					●			◆				◆		

What are the advantages for production scheduling that can be provided by artificial intelligence?

Considering that the PPC function faces changes according to the production environment variations and according to the multidimensional analysis results, the advantages are related to the structuration of an appropriate, tailored, and systematic method for scheduling capable of achieving any of the scheduling objectives (i.e., reducing the makespan). From the information gathered, we could understand that each analyzed system was considered quasi-unique due to the database search constructs (i.e., 'industrial application'). Hence, the approaches used considered the restrictions each system showed naturally, and in this case, AI techniques provided versatility for delivering tailored solutions. Considering this trait, we noted that it was possible to test the system's behavior under different conditions, unexpected events, and multiple constraints to obtained conclusions related to the production scheduling function. This characteristic shows an actual advantage over other techniques since they are closer to real-world manufacturing scenarios. Besides, we could observe that it was possible to apply structured AI methods to any manufacturing system without restrictions. Also, we found that AI approaches have general characteristics that can be transferred when analyzing a system.

From the included articles, all of them reported different benefits in terms of less human intervention, faster methods, and a goodness of the solution for the analyzed manufacturing system when solving the scheduling problem. So, we concluded that there is a notable advantage in solving this problem through AI approaches.

Does the integration of production scheduling with AI determine performance improvements?

The integration between production scheduling and AI determines performance improvements since these approaches can be designed to achieve one of the scheduling objectives (or more than one). From a managerial point of view, it was possible to observe how applying a strategy to solve the scheduling problem showed improvements in a multi-dimensional correlated way. This means that improving one dimension from the set of performance variables carried improvements in other dimensions. For example, most of the analyzed study cases targeted a makespan reduction where the AI approach used in each case could always provide a better result than their initial situation. Therefore, it permitted to attached improvements related to costs, lead time, resource usage, and profitability, dimensions straightly correlated from a management perspective. However, it is important to consider the cost attached to the development, usage, and maintenance of this type of solution and its feasibility. For instance, it is essential to consider the benefit/cost ratio given by investments in computational power to obtain a solution in a feasible time window. So, after the analysis made over the included articles in the SLR, we could conclude that the integration between AI and scheduling implicated performance improvements for each studied manufacturing system.

3.5 Bibliometric analysis

Considering the information from the SLR, we performed a bibliometric analysis from the meta-data generated. We could find information about co-occurrence terms, co-occurrence keywords, use of keywords, productivity and cites per country, and the authorship network. For the generation of maps, we used VOS-viewer software.

Figure 11 shows the co-occurrence of terms or concepts in the content of the included documents in the SLR. It shows the different relationship between these terms and their usage frequency of at least ten times. For our case, we found that the most used terms were 'system', 'product', 'control', and 'scheduling problem'.

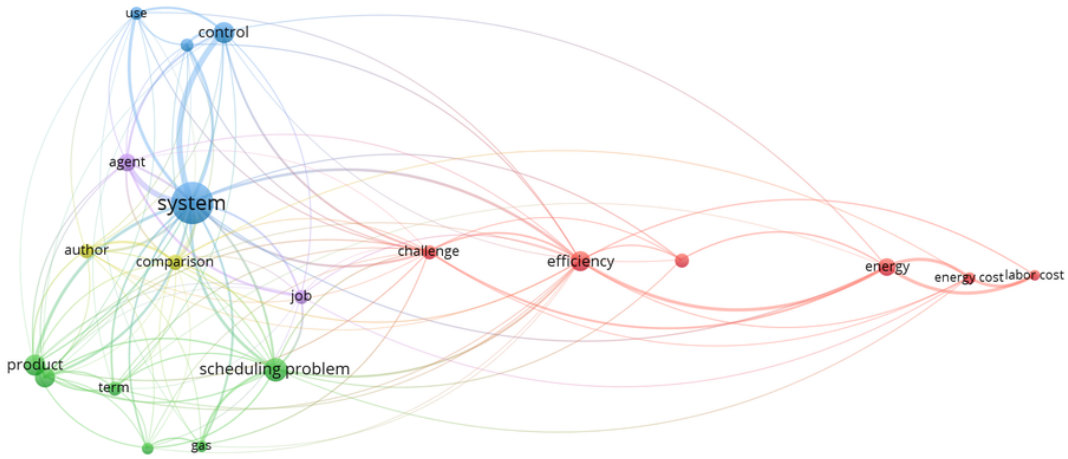

Figure 11. Term's co-occurrence map.

Figure 12 shows the co-occurrence of terms but focused on the evolution in time from 2010 to 2020. Once again, we included terms that were mentioned at least ten times

in a document. The figure shows a tendency to develop solutions for the scheduling problem in the latest years, considering resources as energy and workforce, and its related cost.

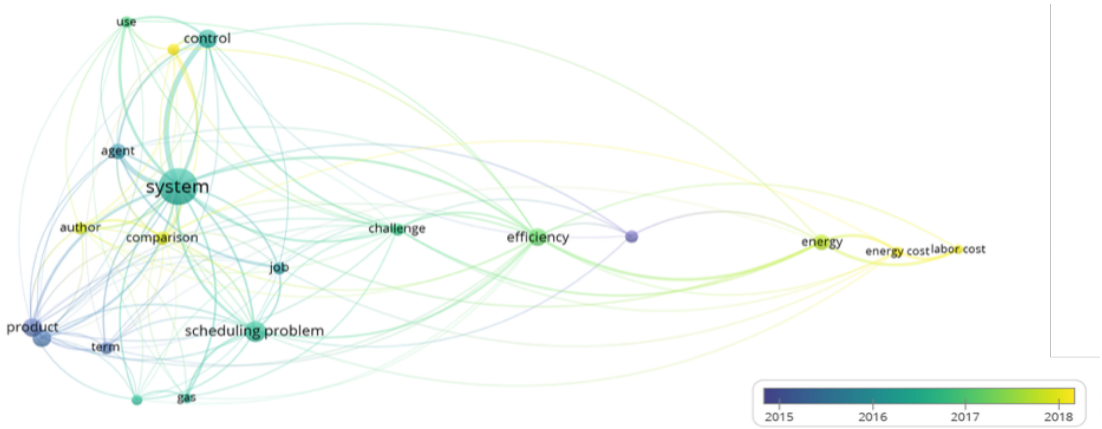

Figure 12. Term's co-occurrence time-line map.

3.5.2 Keyword's co-occurrence map

Figure 13 shows the co-occurrence of the keywords used in the included documents for the SLR. From this map, it is possible to obtain keywords related to the searched topics of this project, which are frequently used. The resulting map highlights keywords as 'production control', 'production control', and 'genetic algorithms'. Finding 'genetic algorithms' is expected because, in almost 80% of AI approaches used, we found the genetic algorithm topic.

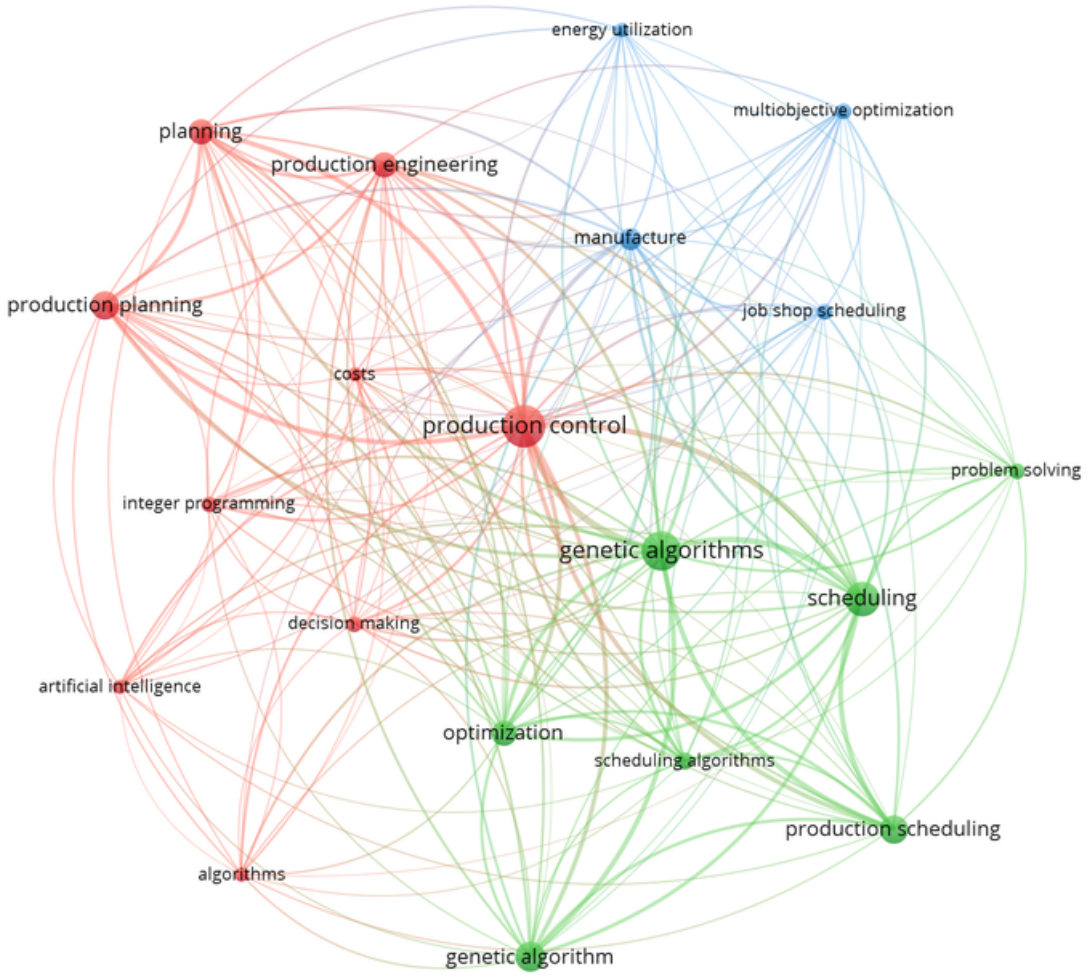

Figure 13. Keyword's co-occurrence map.

3.5.3 Use of constructs

We also analyzed the time trend of keywords usage related to the construction we used in the databases. Figure 14 shows the tendency for the construction related to the AI approaches. It is possible to see how GAs have been used in most cases, something

expected. Still, it is more interesting to see how ML method are becoming a new trend in the latest years.

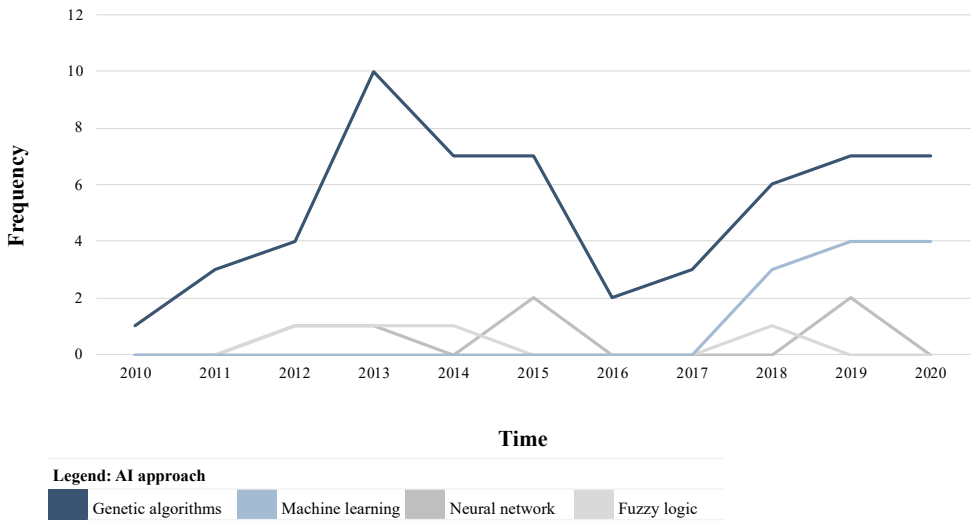

Figure 14. Time trend for the construction related to the AI approaches.

With this analysis, it was also possible to observe how the terms related to planning and control changed in time. Figure 15 shows that production scheduling is becoming preferred since it is a term related to flexibility, a problem complex to solve in real-world applications.

Lastly, we performed the analysis for the constructions used related to applicability. Figure 16 shows the resulting trend from the SLR results. In the figure, it is possible to observe knowledge transfer from the academy to real applications in industry in the last four years. This result is a good precedent for developing projects that aim at real-world

problems since it indicates the existence of literature sources, we can exploit to develop tools applicable to real-world manufacturing systems.

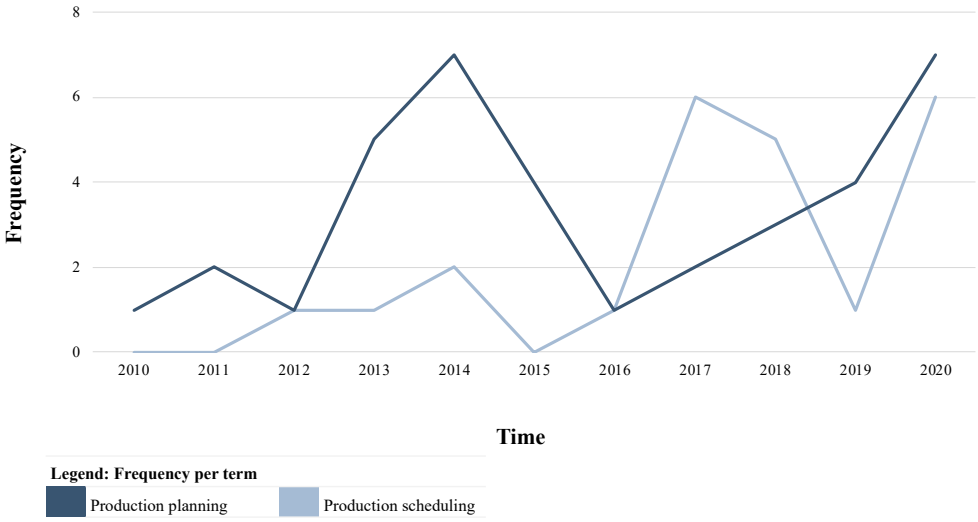

Figure 15. Time trend keywords technicity.

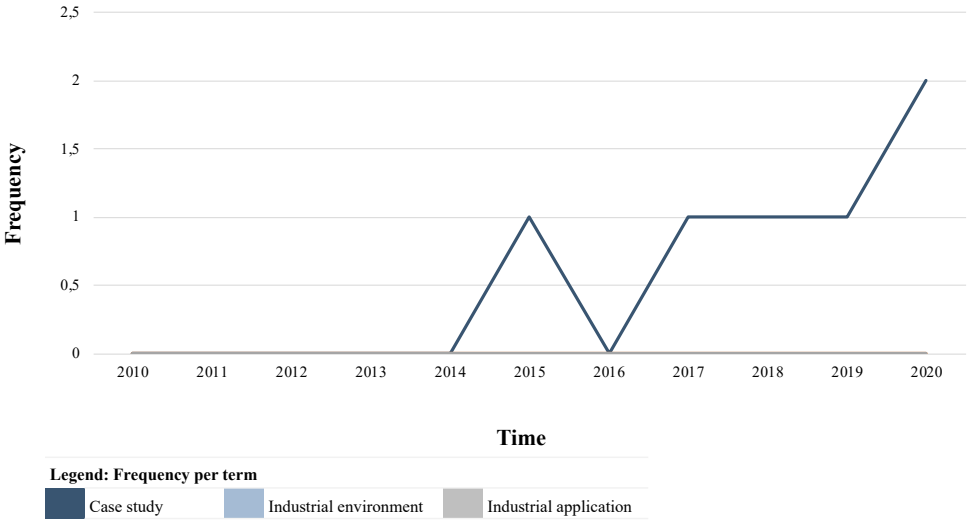

Figure 16. Time trend keywords applicability.

3.5.4 Productivity per country

We analyzed the productivity of knowledge related to the solving of the scheduling problem through AI methods. We found that China, United States, Iran, and United Kingdom are the world's leaders. For us, the result was not unexpected since the countries in this set are economic potencies or highly industrialized countries, where the applicability of AI as a tool is possible. Figure 17 shows the percentages for knowledge productivity around the world based on the included articles of the SLR. Regarding cites per country, China, the United States, Iran, and Turkey are the world's leaders. Once again, we could observe highly industrialized nations that face needs in their production systems solvable through AI methods. Figure 18 shows the results for cites per country.

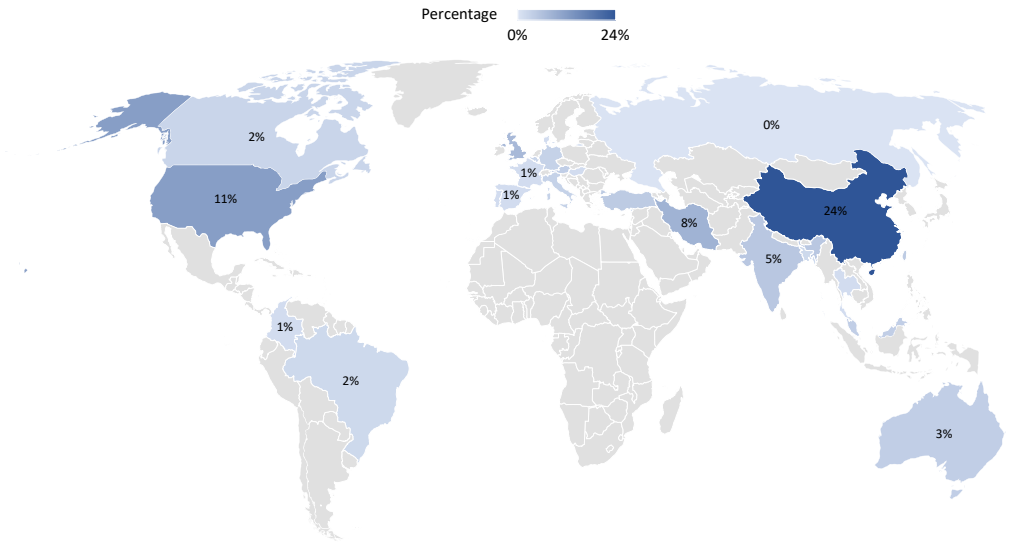

Figure 17. Productivity per country.

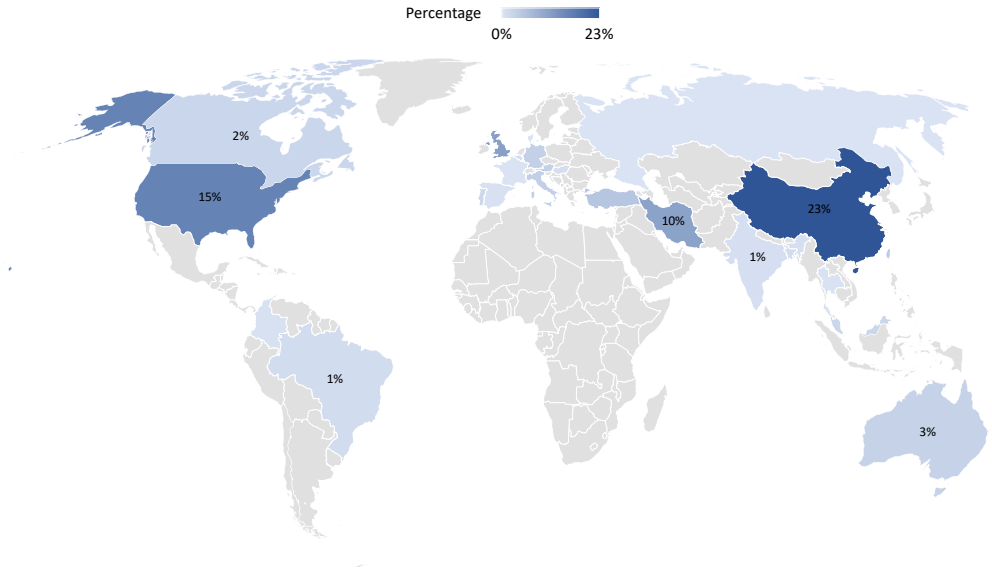

Figure 18. Cites per country.

3.5.5 Co-authorship network

The last analysis we performed from the meta-data generated by the SLR was the co-authorship network. Figure 19 shows how Chinese authors dominate the applicability of AI methods for solving the scheduling problem and how robust their co-authoring is for research collaborations. This result is attached to productivity and cites per country, showing how important China is in this knowledge field. Indeed, figure 16 shows a Chinese author as the core of the co-authorship network.

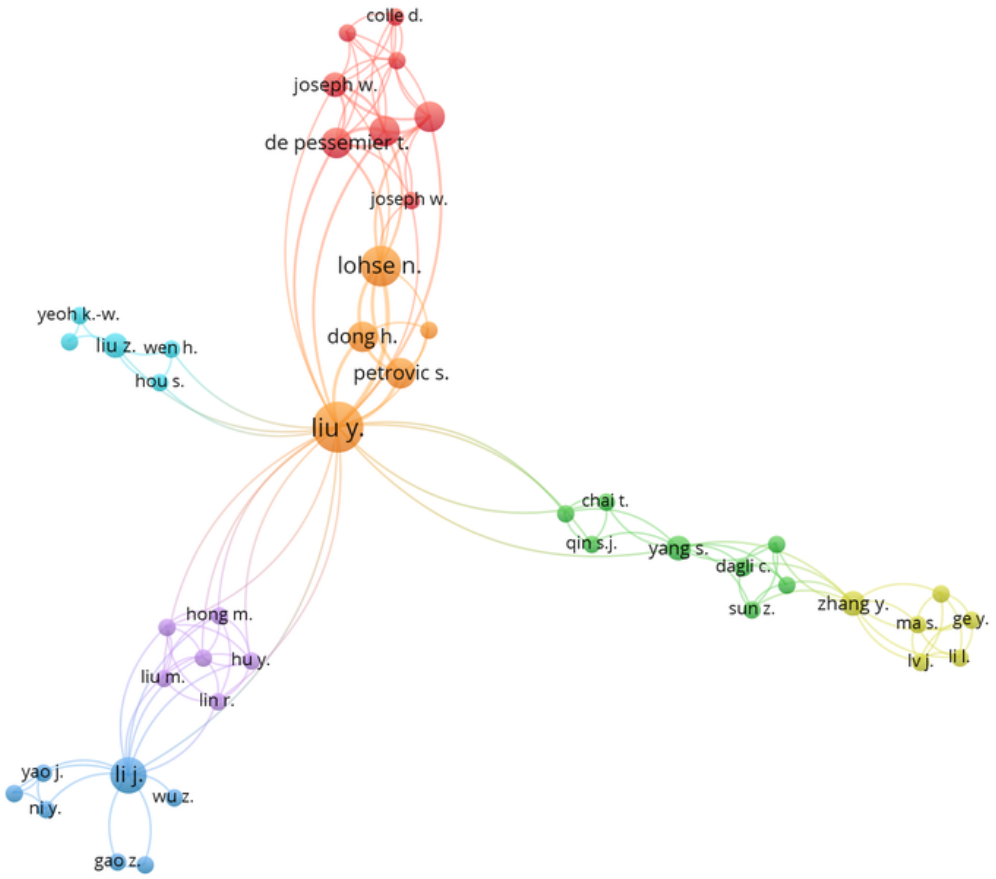

Figure 19. Co-authorship network.

Chapter 4 - SLR Results Discussion

This chapter presents the discussion of the results from the SLR. In this manner, it contains a general discussion about the found gaps and trends. Then, from the critic analysis, we obtained the AI approach most suitable for developing an alternative solution to the scheduling problem and covers some of the SLR gaps. We supported that decision by obtaining information from literature about industry cases where scheduling problem were solved through a ML technique. In the final part, we set the objectives developed in this thesis.

4.1 General discussion

Considering the SLR results, and the meta-data analysis performed, we found that there are trends and gaps when solving a scheduling problem using AI methods. On one hand, the trends:

- Scheduling problem solution for multiple machine systems is more attractive for industrial applications.
- From the objectives to solve in the scheduling problem, reduction of the makespan is frequently studied.
- The studies for solving the problem in a manufacturing environment aimed to improve process performance instead of product performance.

- Given the uniqueness of each case study, it was more frequent to find AI scheduling solutions developed through coding using a programming language than using a software.

On the other hand, the gaps:

- Solution for the scheduling problem in real-world manufacturing systems has not been explored enough through ML techniques (6% of the analyzed studies in the SLR were developed using a ML technique).
- Lack of flexibility for the solution considering the changes that the scheduling process faces in a real-world situation.

Given the gaps and trends, we could consider adopting a different approach from GAs for solving the scheduling problem. It represents a new way to face the scheduling function changes due to today's market demand (i.e., smaller batches, diversified production requirements, etc.). Besides, from the bibliometric analysis made for AI frequency usage, ML is becoming a promising method to solve the same problem. Regarding the system, we could observe no specific restrictions for applying AI approaches to a production system. We found continuous and discrete production systems, dynamic, static, industry, and at the laboratory level. We also found systems made by single and several machines and applicability to a broad range of industry fields. So, there is room for developing a solution applied to a real-world problem, and available information to do so.

Furthermore, there is a gap in flexibility, which we consider essential due to nature's real-world production system. So, we can explore a solution that considers the needed flexibility. From the evaluation through performance analysis, we could conclude that the main objective is reducing the makespan for the production system. Once it is achieved, it drags a reduction of lead time, costs, and increment of profitability. So, we can follow this tendency and aim to decrease the makespan.

In general, we concluded that relating scheduling and AI is beneficial. It brings a significant advantage for production systems in structured, replicable, and standardized methods for solving the scheduling problem. Among the techniques included in the SLR, and considering the existing gap in flexibility, we selected ML as an AI approach that could cover it. Additionally, it can solve in an innovative, not-enough explore way the scheduling problem.

4.2 Machine learning approach

According to the discussion presented above, this subsection presents the most relevant contributions made by different authors for solving the scheduling problem through ML. We classified the contribution into two types: i) indirect solution, and ii) direct solution. The first type relates to those contributions developed for enhancing a previously established solution. For example, when they used a ML technique for improving the performance of a GA-based solution. The second type relates those contributions where the authors developed a solution based on a ML approach.

4.2.1 Machine learning as indirect solution

Gannouni, Samsonov, Behery, Meisen & Lakemeyer (2020) analyze how neural combinatorial optimization (NCO), an emerging field of research, can be applicably scalable in a short-term production planning context through an evaluation metric based on stability and quality of solutions. The purpose of the authors is to propose the usage of NCO for solving combinatorial optimization (CO) problems presented on production scheduling with sequence-dependent setup costs. To prove their statement of solving its problem on scheduling, they made an experiment of setup waste minimization for a blown film extrusion process. In this experiment, the authors define the blown film extrusion problem as a multiple traveling salesmen problem (mTSP) to find the optimal sequence of the items produced, minimizing the total setup time/cost. This problem's constraints were modeled by the capacitated vehicle routing problem (CVRP) that represented the demand that must be satisfied and the limitation machines' capacity. Due to its large size of data, the computation of the total setup time/costs was done by a supervised ML model (Gradient Boosting Machine Regressors) to predict the different setup waste/cost caused by the change of some type of products in a machine. This method presented a variance of 77.1%.

After creating a setup cost function, the authors employ reinforcement learning (RL) for solving the optimization problem, which is modeled as CVRP. The state was defined as the nine most discriminative product features used in predicting the setup waste. Later, the reward was related to the cumulative setup waste's negative total value for the developed order sequence. Besides, the authors used three different training data sets for

the training to evaluate some performance indicators as optimization, robustness, and scalability of the methods. The NCO showed a higher optimality gap compared with Google OR-Tools in all its different training data set. In this sense, the authors compare the efficiency in terms of computational time between exact (Google OR-Tools) and approximate (NCO) algorithms and how the gap between them increases when the problem size increases.

To conclude, the authors considered that this paper's most important contribution is to achieve a complex cost function given by a supervised ML model and a training process with superior time-performance compared with other solvers.

He, Li, Zhang & Cao (2019) proposed a discrete multiple-objective fireworks algorithm (DMOFWA) to solve a mathematical model for a multiobjective flow shop scheduling problem with sequence-dependent setup time (MOFSP-SDST). DMOFWA is a new swarm intelligence algorithm unexplored in multiobjective optimization problems. It requires using ML techniques such as clustering analysis (CA) and opposition-based learning (OBL) to improve the swarm intelligence algorithm's performance through the study of its historical data. The OBL helps the algorithm learn from the current search area and generate opposite solutions in the opposite unknown search area. In this way, this opposite solution's quality resulted better than the initial solution proposed. Then, CA helped to cluster those similar solutions into the same group. As part of the development of this paper, the authors built a mathematical model based on MOFSP-SDST. They

developed a multiobjective approach based on fuzzy correlation entropy analysis (FCEA) for that specific mathematical model. Moreover, the authors proposed integration between DMOFWA, OBL, and CA. Finally, they validated the effectiveness of the DMOFWA by experiments, comparing different performance indicators such as hypervolume metric, spread metric, and C metric that reflect convergence, diversity, and distribution uniformity, respectively. The DMOFWA performed better in 8 out of 11 tests in terms of hypervolume metric. In the spread metric, this algorithm performed better than the others in 7 out of 11 trials. Therefore, the authors demonstrated that the DMOFWA proposed was efficient using ML methods, outperforming MOLSD (multiobjective local search based decomposition), MODABC/D (multiobjective discrete artificial bee colony algorithm based decomposition), MOILS and RIPG.

Zhao, Gao & Li (2019) explored the scheduling problem from a flexibility point of view. For their research, the authors considered machine failures as a disturbance in production scheduling. They showed that different rescheduling techniques had been proposed to deal with this type of event. Their performances depend on the status of machine failures, with no single strategy for every failure status. Considering the possible advantages that AI and ML bring for the scheduling function, the authors proposed a new rescheduling decision model based on random forest, a ML method, to learn the optimal rescheduling strategy in different machine failures. In this way, they set the scheduling problem as a dynamic job shop rescheduling problem. So, based on ML, the solution

selected the best rescheduling strategy from a group. In this case, the solution based on AI evaluated the impact of a machine failure on an original schedule. They considered two indicators: the delay and deviation. According to the prediction made for the failure's consequences, it could choose one of the available rescheduling strategies. Thus, they demonstrated that the internal mechanism between machine failures and rescheduling strategies could be learned through a method. As a significant conclusion related to AI techniques' contribution to scheduling, the authors expressed that their research's contribution was the establishment of a new model that combines ML techniques and traditional scheduling approaches.

Scholz-Reiter, Heger & Hildebrandt (2010) developed an approach for selecting dispatching rules considering the system conditions, which set a performance dependency for these rules on different scenarios. Since Gaussian processes (GP) have outperformed other methods, the authors proposed using them to select dispatching rules in dynamic scenarios. The type of problem addressed was a dynamic shop scenario with high complexity. The arrange of dispatching rules used were: shortest processing time (SPT), earliest due date (EDD), first in first out (FIFO), modified operation due date (MOD), and 2Processing time + work in next queue + next processing time (2PTPlusWINQPlusNPT). The mean tardiness was selected as a performance criterion. The obtained results through GP learning were of high quality, increasing when the available data increased. However, even a few data points (10 to 20 data points) were sufficient to improve over the best simple

choice of always using MOD. With 30 data points, it was possible to achieve 91.5 % of the improvements possible with switching dispatching rules based on the two attributes utilization and due date tightness. Using 180 data points yielded up to 99.9 %. In this way, they proved that scheduling with dispatching rules could be further improved. Besides, it was possible to adapt the scheduling mechanism to changes by selecting the best rule for a given system status through AI techniques.

4.2.2 Machine learning as direct solution

Hubbs et al. (2020) examined the application of RL to a chemical production scheduling process under uncertainty to achieve online, dynamic scheduling and benchmarks the results with mixed-integer linear programming (MILP) model. In their work, they used RL to address the uncertainties in the production planning and scheduling problem and illustrate its application in an industrial, single-stage, continuous chemical manufacturing process. The scheduling algorithms' goal is to build a production schedule for the entire planning horizon, while the purpose of the reinforcement learning, and optimization methods is to maximize the site's profitability over the simulation period. Five types of models were developed for comparison: a deep reinforcement learning (DRL) model, deterministic MILP on a rolling horizon, a deterministic MILP on a shrinking horizon, a stochastic MILP on a rolling horizon, and a perfect information MILP that optimizes over the entire horizon and without uncertainty. The DRL model is trained using the advantage actor-critic algorithm (A2C). This DRL approach is known as a model-free

algorithm, meaning there is no transition function of the system that is given to the RL agent. It must learn a good policy through trial and error.

For this reason, the model must be trained extensively by making mistakes early on and then learning what actions yield high rewards over time. As scheduling results, they showed that the RL approach led to better product availability levels than other methods. Nevertheless, since the product availability is higher, so are the inventory costs, which provide a buffer in the face of uncertainty.

The computational cost when using a RL method is paid upfront, while each additional inference during testing is computationally cheap. To simulate one training episode, the RL agent takes about 0.8 s, but it must do this thousand times. For inference, the time is greatly reduced because the RL system only needs to make decisions for one episode and is not required to update the weights via the computationally intensive backpropagation algorithm. On the other hand, the MILP algorithms only need to solve each scenario one time but solving 90 smaller sub-problems as the simulation runs through time. Regarding robustness, the authors showed that the RL approach remains relatively consistent in its scheduling decisions when the environment changes (i.e., demand) compared to the other methods. However, since it is model-free, the RL method has fewer capabilities when responding to significant environmental changes. These limitations can be addressed by re-training the system on the new distribution before deployment or developing hybrid models.

Riemer-Sørensen & Rosenlund (2020) examined the usage of DRL (soft actor-critic algorithm) in long-term hydropower production scheduling. They aimed to optimize the revenue per year, which was impacted by storing water for future power production with an unknown electricity price or producing power immediately with a spot price of electricity. To achieve this, the authors define the actor-critic algorithm (environment, action, reward, and state) and a new network called "target value" used as a memory of how the value network was in the immediate past. Concerning its hyperparameters, they work with RMSprop to be more adaptable for non-stationary data distribution and avoid dependence on momentum as Adam optimizer. Moreover, they use a lower learning rate to explore the system instead of exploiting the experience gained. To validate it, they employ a case study in which they took one day to train the algorithm using 300000 episodes. They demonstrated that an actor-critic algorithm is ideal in stochastic environments, given its stability. To sum up, the authors suggest a high potential for RL when it comes to disaggregate models in cases in which the entire system is too complex and computationally heavy for a single model. Therefore, this method's flexibility in complex scenarios can complement the current optimization methods in hydropower scheduling.

Nie, Wang, Bai & Wu (2019) explored the scheduling problem for a dynamic job shop, where jobs may arrive over time, and their arrival dates cannot be predicted in advance due to different reasons (cancelation, transportation, machine breakdown, etc.). In

this manner, they developed a ML approach based on gene expression programming (GEP) to solve dynamic job shop scheduling problem (DJSSP). In their work, the objective was to minimize the order's average flow time through the system. As a result, they obtained that the proposed approach, which is enhanced through a ML tool, is more effective than the human-made rules and slightly better than GEP. In this way, the proved the benefits coming of solving the scheduling problem through an AI approach.

Kuhnle, Schäfer, Stricker & Lanza (2019) explored implementing a RL technique to solve the scheduling problem for a manufacturing system. In their research, the authors introduced a systematic approach and guideline for designing, implementing, and evaluating RL algorithms for adaptive order dispatching. In this way, they explored the advantages that AI approaches can provide to scheduling in terms of flexibility. The AI approach applied was a blended method between Q-learning and an artificial neural network (ANN), while the manufacturing system considered was a job shop. They chose a blended approach since they considered the computational cost associated to a technique as Q-learning. As a result, they obtained machine utilization and lead time, and then they compared them with the values obtained when the problem is solved through heuristics. They concluded that the reward behavior showed that the used approach achieved a good learning performance. They also showed that machine utilization and lead time can be optimized through the technique used, which is not easy to accomplish since both objectives are contradictory. Moreover, they showed that compared to a heuristic

dispatching rule, the approach proposed demonstrated a better result of the adaptive control.

Considering the market evolution, Waschneck et al. (2018) presented a production control method that satisfies the requirements related to extensive portfolios with small quantities of each product. To increase production control performance, the authors applied DRL to production scheduling in complex job shops utilizing cooperative deep Q network (DQN) agents. Besides, through DQN, it was possible to ensure balance for different objectives. They considered flexibility, global optimization, and automation. However, the computational cost that represented the development of the tool was considered a disadvantage. The manufacturing system studied was a complex job shop composed of four work centers. As result, they obtained that, compared to dispatching heuristics, there was a reduction in the distribution of the cycle times for the same type of product up to 50%. They also obtained that the ratio of the lots with severe delay decreased tremendously. For product type 1 they obtained a reduction of 92%. For type 3, they obtained a reduction of 86%. As the main conclusion, the affirmed that the system was able to develop a global optimal scheduling solution without human intervention or prior expert knowledge. Besides, it can be trained and exchanged within hours.

4.2.3 Remarks

We focus our interest on using a ML approach as a direct solution for the scheduling problem. Taking the information presented above, RL was used when the set problem involved flexibility to face changes in the system and a reduction in the computational cost. Hubbs et al. (2020), Riemer-Sørensen & Rosenlund (2020), Kuhnle et al. (2019), and Waschneck et al. (2018) established the need to include flexibility for their proposed solutions, considering a feasible computational time since their solutions were meant to be applied into real-word industry cases.

On the other hand, for the studies that improved a previous solution through ML, the most frequent approximation was the supervised based. This type of ML approach permitted to include flexibility to the solution since it allowed the prediction of events that triggered changes in an initial schedule (Gannouni et al. (2020); Zhao et al. (2019); Scholz-Reiter et al. (2010)). Exploring these types of solutions is not of interest in this thesis, but the conceptual framework is useful for contextualization and comparison.

4.3 Research objectives

4.3.1 Motivation

As discussed in the introduction, the PPC function (including scheduling) faces continuous updates due to changes in production. These changes, which represent a more challenging environment, are related to the market's requirements, the introduction of new products, production asset breakdowns, lack of demand, etc. In this way, companies rely

on different information and communication technologies tools to face these changes and keep being competitive. Considering this relationship, we based the motivation for developing this thesis in three dimensions: i) results from the SLR, ii) technical perspective, and iii) managerial perspective.

Results from the SLR

The results from the SLR were presented at the beginning of this chapter. In summary, the advantages of using AI approaches for solving the scheduling problem are related to the structuration of an appropriate, tailored, and systematic method that permits solving the problem and achieves an objective. Besides, because of how AI approaches are being applied for solving the scheduling problem, they provide versatility in terms of adaptability to different production systems and objectives to achieve. This characteristic is a determinant for industry applications since it is possible to find complex manufacturing systems in real-world manufacturing companies. For example, one company can have transfer lines and job shops in its production system (Ruozi, 2020). In this manner, using AI approaches for solving the scheduling problem could represent a notable advantage.

Furthermore, we stated that using AI methods for solving the scheduling problem determines performance improvements because good scheduling represents less cost, less throughput time, less use of resources, more profitability, etc. (Hubbs et al., 2020). However, developing a solution based on AI approaches can be very complex, which is correlated to the number of objectives to achieve, the system's features, and the goals for

the objectives. So, it is essential to consider that the greater the problem's complexity, the higher the cost (i.e., computational resources).

The SLR showed that, among the AI techniques, and considering the existing gap in flexibility, a ML-based approach is the best option for developing an alternative solution to the scheduling problem. Because of the literature's information, RL was used when the set problem involved flexibility to face changes in the system and a reduction in the computational cost, so we selected it as a solution approach for this thesis.

Technical perspective

From a technical perspective, using AI for scheduling positively affects the production system's performance due to human intervention's lesser dependency for activities with low added value and ease to automate. Besides, considering changes that the industry will face in the future, such as customized products, changing demand, flexible/reconfigurable manufacturing systems, and lower reaction times, AI deals with managing those changes through accurate forecasting, data analysis, and simulation. In the industry 4.0 context, complex production systems as parallel machines with intermittent arrivals are expected to be required in terms of flexibility, so it is needed to focus the attention on developing solutions capable of scheduling tasks in these dynamic environments. AI could be the enabler for solving the scheduling problem in a constantly changing environment.

Managerial perspective

From a managerial perspective, the scheduling process is a cost determinant for a company since it assigns tasks to resources. Costs like time, raw material, labor, and machine usage are related to the scheduling process. In this way, better use of resources through efficient production management means saving costs for the company. Furthermore, good management of the scheduling process can represent a competitive advantage, which provides differentiation and a better service level to those demanded by the market.

4.3.2 Objectives

Considering the motivation, we set as main objective the developing a scheduling tool for a complex manufacturing system through a solution based on a ML approach. As a first activity to achieve the main objective, we considered the execution of a literature review regarding AI techniques for solving the scheduling problem. Therefore, it was necessary to research the different scheduling problems on various production systems and their specific characteristics to have enough technical support to develop the tool. As a second activity, creating the tool that solves the scheduling problem for the analyzed system, and the respective validation to demonstrate its performance. Having said that, the general objective of this thesis was:

"Development of scheduling tool based on AI approach using (static) process-time based scheduling techniques and actor-critic as RL method applied on a parallel machines production system with intermittent job order arrivals."

Regarding what was required to achieve this objective, we employed mock data to train the tool. The code and the experiments were developed in Python, since it is a general-purpose, open-source tool, and today it is the most adopted programming language in the world.

Practical objectives

For achieving the main objective of this thesis, it was necessary to stablish a set of specific objectives considering three axes: i) the development of a tool, ii) consideration of the simulation time, iii) comparison between other solutions.

- **Tool:** It was the mean through which the scheduling problem was solved. Besides, it was the instrument where the tests were run, and the results were shown. As this thesis explored a RL approach to solve the problem, it was necessary to build the tool using the information collected from the SLR. It was needed to analyzed studies with similar manufacturing systems to the one we pursued to obtain the theoretical background that the authors applied to solve the problem.

- **Simulation time:** Since the alternative given by this thesis provided a solution to the problem established in the main objective and considering that there are exiting exact methods to solve those kind of production systems in a static environment, the tool's simulation time was a parameter to be considered. As this thesis aimed to solve the scheduling problem thinking in a real-world manufacturing system, we needed to consider the tool's time for providing a solution. In this way, considering that companies schedule and reschedule between working shifts, the tool's response time was very relevant.

- **Solution evaluation:** Considering that this thesis explored a RL for solving the scheduling problem, it was important to compare the obtained results with other methods reported in the literature. To make conclusions about the feasibility of the provided solution, we needed to compare resulted performances, simulation time, and managerial performance.

Chapter 5 - Theoretical Background

This chapter presents the theoretical background needed for achieving the objectives of this thesis. The first part presents the information necessary for understanding the scheduling problem. So, it contains information about production system environments, constraints on scheduling problems, and scheduling techniques. The second part shows the definition and theory behind AI. In this way, it contains what AI is, the theory about ML approaches, RL, and actor-critic (A2C) method explanation.

5.1 Production scheduling

5.1.1 Notation

According to Pinedo (2016), the general notation in the scheduling problems considered here are:

$j = \text{job} \in \mathbb{R}^+$

$i = \text{machine} \in \mathbb{R}^+$

$n = \text{number of jobs} \in \mathbb{R}^+$

$m = \text{number of machines} \in \mathbb{R}^+$

$(i, j) = \text{processing steps or operations of job j on machine i}$

And the variables associated with job j are:

- $\boldsymbol{p_{ij}}$**:** Processing time. It represents the processing time of job j on machine i. If the job j does not depend on the machine i or if the job j can be only processing on one specific machine, the notation i could be omitted.
- $\boldsymbol{r_j}$**:** Release date. It is the date in which the job j arrives to the system.
- $\boldsymbol{d_j}$**:** Due date. It is the arranged shipping or completion date. It is allowed the completion of a job j after its due date, but it could be penalized. If the due date must be done, it is possible to call it $deadline$ and it is denoted by $\overline{d_j}$.
- $\boldsymbol{w_j}$**:** Weight of job j. It is a priority factor, in which it is denoted the importance of a job j among the others in the system.

The scheduling problem could be described by three fields $\alpha \mid \beta \mid \gamma$. The machine environment is given by the α field and it has just one entry. The β field is related to the characteristics of the process and its constraints; in this field, the number of entries could vary. The γ field describes which objective will be minimized; often it will contain a single entry.

5.1.2 Production system environment

The production system environments are defined by the relationship among jobs and machines. Pinedo (2016) defines the following production systems:

- **Single machine:** This case is the simplest one. It helps as a base for all the other machine environments that have a higher degree of complexity.

- **Machines in parallel with different speeds:** In this case, there are m machines in parallel with different speeds on the system (speed of machine i denoted by v_i). If the job j does all its operations in the machine i, then the time to perform job j on machine i will be equal to p_j/v_j. In case that all the machines have the same speed, this environment will be equal to the identical machine in parallel machine environment.
- **Flow shop:** In this case, the m machines are arranged in series. Each job must pass for all the m machines, following a predefined route, and all the jobs must do it. When one of the machines (example: i) finishes a job, the same job goes to the queue at the next machine $(i+1)$. In the queues, the general rule to dispatch the jobs to the machines is $First\ in\ first\ out$ $(FIFO)$, that means that a job cannot goes directly to the machine if there is another job waiting in the queue. If the system does not respect the previous rule, it is called $permutation$ flow shop and it includes the entry called $prmu$ in the β field.

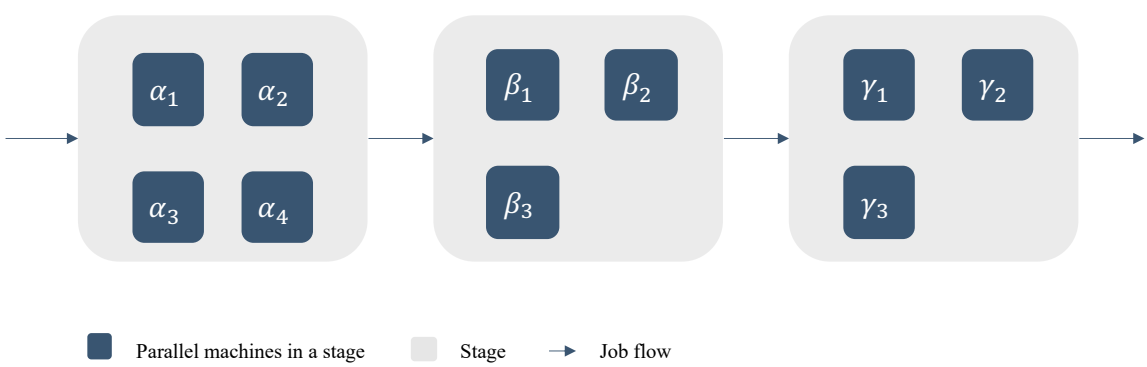

Figure 20. Flow shop diagram for three stages in series.

- **Flexible flow shop:** This is a general case between a flow shop and parallel machine environment. In this case, there are c stages in series instead of machines, and on each stage, there is a group of identical machines in parallel. The process starts when the job j enters to the first stage, and inside of it, the job should be processed on just one of the machines and all of them are able to do it. Between stages, it is possible to have queues that can be regulated by $First\ come\ first\ serve$ $(FCFS)$ rule. The process finishes when the job visits all the stages.
- **Job shop:** In this machine environment, there are specific routes per each job to follow among the machines. There are two types of job shops: i) when each of the jobs visit each of the machines at most once and, ii) when a job could visit each machine more than once. For the last type, the entry $rcrc$ for $recirculation$ will be included in the β field.
- **Flexible job shop:** This is a general case between a job shop and parallel machine environment. In this case, there are c work centers in series instead of machines, and on each stage, there is a group of identical machines in parallel. Following the job shop nature, each job will have its own route on the shop. Therefore, the job j needs to be processed at each work center but on just one of the machines, considering that all of them can process it. In this machine environment is possible to find the entry $rcrc$ for $recirculation$, even that the shop could be visited more than once for a job.

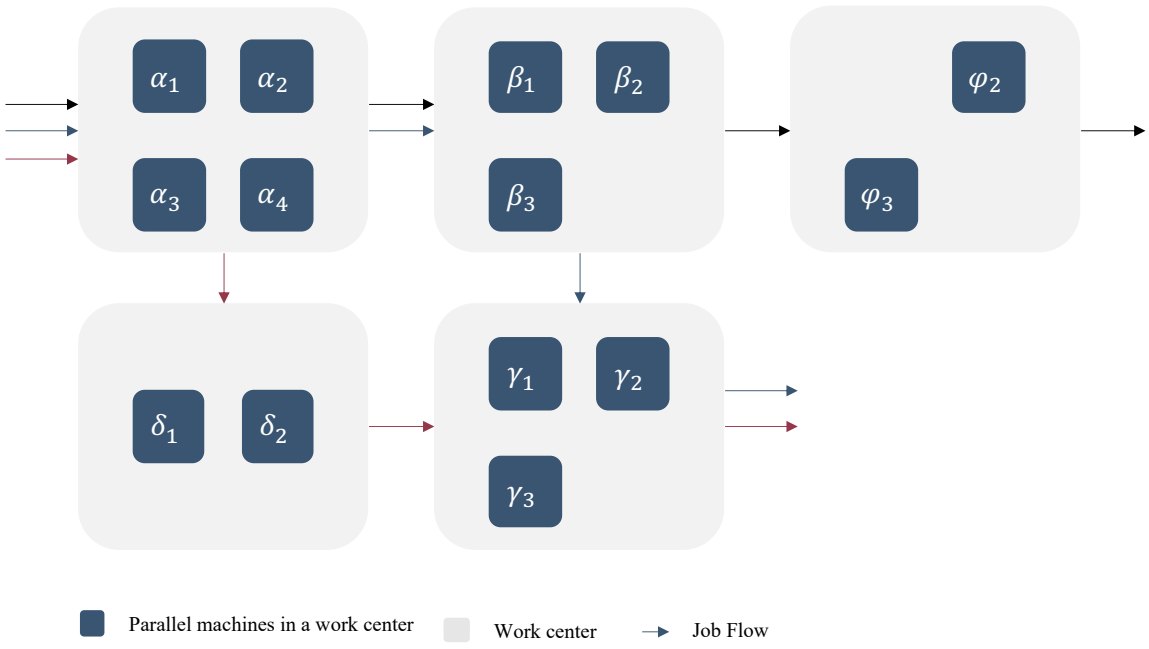

Figure 21. Job shop diagram with five work centers and three jobs with different routes.

Given the definition for flow shop and job shop, it is possible to mention some of their characteristics. Table 4 shows a summary of the main features. Regarding production management, it relates the scheduling and controlling issues possible to find in those types of systems. The flow shop is simpler than a job shop because this system should deal only with two types of problems: batch size and batch sequence. Instead, a job shop must deal with scheduling problems related to loading and timing. In a job shop, the scheduling shifts to a multi-objective decision-making problem with goals often in conflict, such as increasing the machine utilization and achieving delivery dates of job orders.

Table 4. Flow shop and job shop characteristics.

Production system	Characteristics				
	Production management	Machine utilization	Impact of failures	Lead time	Flexibility
Flow shop	Simple	High	High	Low	Low
Job shop	Difficult	Low	Low	High	High

For machine utilization, the job shop shows a lower value than the flow shop. This deficiency for a job shop is given by problems related to changes among product families. This conflict is not typical for flow shops since it usually works with one product family per arrange. Due to its serial configuration, the flow shop system is more affected by the machines' failures because it causes different events as blocking (upstream) or starvation (downstream) between stages. On the other hand, machine breakdowns have a lower negative impact in a job shop since the work centers could serve different product families with different routes, avoiding those work centers that are not available.

Lastly, for flexibility, it can be measure in two different parameters: production mix and modularity. A flow shop system is limited on production mix because it is designed to produce batches for the same job or the same family of jobs (clusters). Moreover, it is possible to observe that in high demand variations, flow shop systems require a

reconfiguration that carries new costs; therefore, the volume flexibility is limited. Instead, the job shop offers higher flexibility in terms of product mix, but it needs to lead with lower production volume. Furthermore, modularity is a strength in this type of systems since it enables the reconfiguration in case of job variations, avoiding significant investments.

5.1.3 Constraints on scheduling problems

With respect to the β field, related to the constraints of the system, they are defined as follows:

- **Release dates (r_j):** Variable that indicates if the job j could start to be processed or not. If the r_j variable exists, then the job cannot be processed before that date and if it does not exist, job j could start at any time.
- **Preemptions ($prmp$):** This constraint allows to the scheduler to interrupt the current process of a job at any time and replace it with a new job on the same machine. That means, the job does not stay on the machine until it is completed. The advantage is the process time cumulated by the job j until that moment is not lost, because the job that was replaced on a specific machine will take the $remaining$ processing time and not the total time to finish the operations of the job j on the machine i. If the preemptions are not allowed, the $prmp$ entry is not included.
- **Precedence constraints ($prec$):** These types of constraints could appear in scenarios with a single machine or in parallel machine environment, in which it is

necessary that one or more jobs must be completed before another job start its process. For the precedence constraints, there are three types:

- If each job has as maximum one predecessor and as maximum one successor, the constraints are considered as $chains$.
- If each job has as maximum one successor, the constraints are considered as $intree$.
- If each job has as maximum one predecessor, the constraints are considered as $outtree$.

The jobs are not forced to accomplish precedence constraints if the entry $prec$ is not included.

- **Sequence dependent setup times (s_{jk}):** This constraint is related to the time that is acquired between a job j and k. If the setup time between job j and k also depends on the machine, the notation should include i as s_{ijk}. If there is no setup time constraint, it is possible to say that setup time is zero or sequence of the jobs is independent, and the entry s_{jk} may be incorporated in the processing time.
- **Job families ($fmls$):** In this constraint, there are n jobs that correspond to a F different families of jobs. The jobs that belong to the same job family ($family\ g$) could have different processing time on the same machine, but they do not require setup times among them. On the other hand, if a new job family ($family\ h$) arrives to the machine, it is necessary a setup time denoted by s_{gh} if there is a sequence

dependence. In case the setup time depends only on one of the families, it could be denoted by s_g or s_h, and if it does not depend on any of them, it should be denoted by s.

- **Breakdowns ($brkdwn$):** This is a constraint related to the availability of the machines due to, for instance, shift changes or preventive maintenance of the machines.
- **Machine eligibility restrictions (M_j):** This constraint could come out when there is a machine environment related to machines in parallel. It is necessary define M_j when a job j can be processed just on specific machines. The notation M_j gives the list of capable machines to do the job j.
- **Blocking ($block$):** This is a phenomenon that happens in flow shop machine environment. In those cases, there is a buffer between two machines arranged on series. If the buffer is full, the upstream machine cannot continue producing the next job, so it will be blocked. In this case, the upstream machine should keep the job on it, to avoid the arriving of a next job that could not be done. Here, it is necessary to apply $FIFO$, that means this phenomenon also include permutation constraint.
- **Recirculation ($rcrc$):** Recirculation entry may happen in job shop or flexible job shop machine environment, in a job can visit a machine or work center more than once.

5.1.4 Objective functions in scheduling

The objective function to be minimized is always related to the completion times of the jobs, hence, to their scheduling. Therefore, in γ field is possible to see objective functions in terms of time. The principal variable to measure in this field is the completion time, and it is denoted by C_{ij}, that means the time to complete the job j on the machine i. In the scenario in which the job j is leaving the last machine in which it should be processed, the completion time is denoted by C_j. This variable gives place to the $Lateness$ and $Tardiness$ variables, denoted by L_j and T_j respectively.

$$Lateness: L_j = C_j - d_j \qquad (1)$$

$$Tardiness: T_j = \max(C_j - d_j, 0) = \max(L_j, 0) \qquad (2)$$

Lateness is a positive value if the job j is finished after the due date (d_j) and it is negative if the job is finished before d_j, while tardiness is always positive. Here, it is possible to penalize the job j with $Unit\ Penalty$ (U_j) if:

$$U_j = \begin{cases} 1 & if\ C_j > d_j \\ 0 & otherwise \end{cases} \qquad (3)$$

The possible objective functions can be the following:

- **Makespan (C_{max}):** Defined as $C_{max} = max(C_1, .., C_n)$, this objective function is related to the completion time of the last job to leave the system. This function has a higher impact in the maximization of machines utilization objective function.
- **Maximum Lateness (L_{max}):** Defined as $L_{max} = max(L_1, .., L_n)$, the lateness determines which job is the worst in terms of non-compliance of deadlines.
- **Total weighted completion time ($\sum w_j C_j$):** This function indicates how much inventory costs is possible to have due to a schedule. It is also called weighted flow time.

5.2 Scheduling Techniques

Given that scheduling requires the allocation of resources, some suggested rules assign the jobs with the highest priority among other jobs to the machines. Scheduling methods assign those priorities, according to Waschneck, Altenmüller, Bauernhansl & Kyek (2016).

The scheduling methods are classified in two groups: exact and approximate methods. The exact methods have the objective to find the optimal solution for any specific problem. The drawback is that their efficiency is limited on NP-Hard problems, because if the problem is large, the method requires more time to find the solution, exploring all the solution space. The approximate methods led with this situation and use less time to find a good and feasible solution, that is not the optimal one. Here, it is possible to find the

‘priority dispatching rules’ and they are chosen depending on the objective function of the scheduling problem (Zahmani, Atmani, Bekrar & Aissani, 2015).

The dispatching rules are split in two categories: static and dynamic. Table 5 shows a list of dispatching rules in each category. The static rules are used when the priorities of the jobs do not change within time and usually, they are related to the following parameters:

- **Processing time (PT):** Time needed to operate a job j on machine i.
- **Process sequence (PS):** Total sequence needed to finish job j.
- **Total processing time (TPT):** Total processing time needed to finish job j.

Table 5. Dispatching rules classification.

Static rules	Dynamic rules
First in first out (FIFO)	Earliest creation time (ECT)
Last in first out (LIFO)	Longest creation time (LCT)
Shortest processing time (SPT)	Shortest waiting time (SWT)
Longest processing time (LPT)	Longest waiting time (LWT)
Shortest process sequence (SPS)	Least total work remaining (LTWR)
Longest process sequence (LPS)	Most total work remaining (MRWR)
Shortest total processing time (STPT)	
Longest total processing time (LTPT)	

Regarding to dynamic rules, they are used when the priorities of jobs changes within time. Moreover, its parameters are given by:

- Creation time (CT) of a job j.

- Waiting time (WT) of the job j on the queue of the machine i.
- Total remaining processing time (or total work remaining - TWR) of a job j to be done.

Hereinafter, the definition for some of the dispatching rules given by Law (2006):

- **Fist in first out (FIFO):** The first job which enters to the system leaves it first. Usually, it leads to overall inefficiency, but it is the simplest one.
- **Last in first out (LIFO):** The last job which enters to the system leaves it first.
- **Shortest processing time (SPT):** This rule selects job j that has the minimum processing time on machine i. The objective is to minimize work in progress and increase throughput of the system through finishing small jobs that are simpler to produce. Its drawback is that large jobs are relegated, and they could ignore due dates or arrival times.
- **Longest processing time (LPT):** Increase work in progress and the short jobs are relegated.
- **Shortest process sequence (SPS):** Select the job j which has the small $chain$ counting predecessor and successor jobs.
- **Longest process sequence (LPS):** Select the job j which has the large $chain$ considering predecessor and successor jobs.

In case the job j is processed on more than one machine, it is possible to use:

- **Shortest total processing time (STPT):** This rule selects job j that has the minimum total processing time.
- **Longest total processing time (LTPT):** This rule selects job j that has the maximum total processing time. There is no criterion to apply it on production systems.

Regarding dynamic rules:

- **Earliest creation time (ECT):** It selects job j that was created first and relegating the jobs that were entering to the system when it starts its process.
- **Longest creation time (LCT):** It selects order job j that was created last and relegating the jobs that were in the system when it starts its process.
- **Shortest waiting time (SWT):** Select job j that has the minimum time waiting in the queue of the machine i.
- **Longest waiting time (LWT):** Select job j that has the maximum time waiting in the queue of the machine i.
- **Least total work remaining (LTWR):** Selects job j with the minimum time in the remaining operations, therefore, it is nearest to be completed.
- **Most total work remaining (MTWR):** Selects job j with the maximum time in the remaining operations to be executed.

These rules already presented are process time-based rules, that means they are related with the objective function of minimization of the makespan, lateness, and tardiness that presented previously.

5.3 Artificial intelligence

5.2.1 Agents

The term intelligent agent is not new in AI. Agent denotes a system that processes information and produces an output from an input. These agents may be classified in many ways. In computer science, software agents are employed to calculate a result from user input. On the other hand, in robotics, hardware agents (autonomous robots) are used, which have sensors and actuators at their disposal. The robotic agent can perceive its environment using sensors, whereas it carries out actions and changes its environment with the actuators (Ertel, 2017). Figure 22 shows schematically the concepts described before.

Ertel (2017) stated that there is a classification between reflex agents and agents with memory. Reflex agents only react to an input. Contrary, agents with memory include the past in their decisions. For example, a driving robot that senses its position and time, as a reflex agent, has no way to determine its velocity. However, if it saves its position at discrete time steps, it can surely calculate its average speed for each of the registered time intervals. Reflex agents are sufficient in cases where the problem involves a Markov decision process. This is a process in which only the current state is necessary to determine the optimal next action.

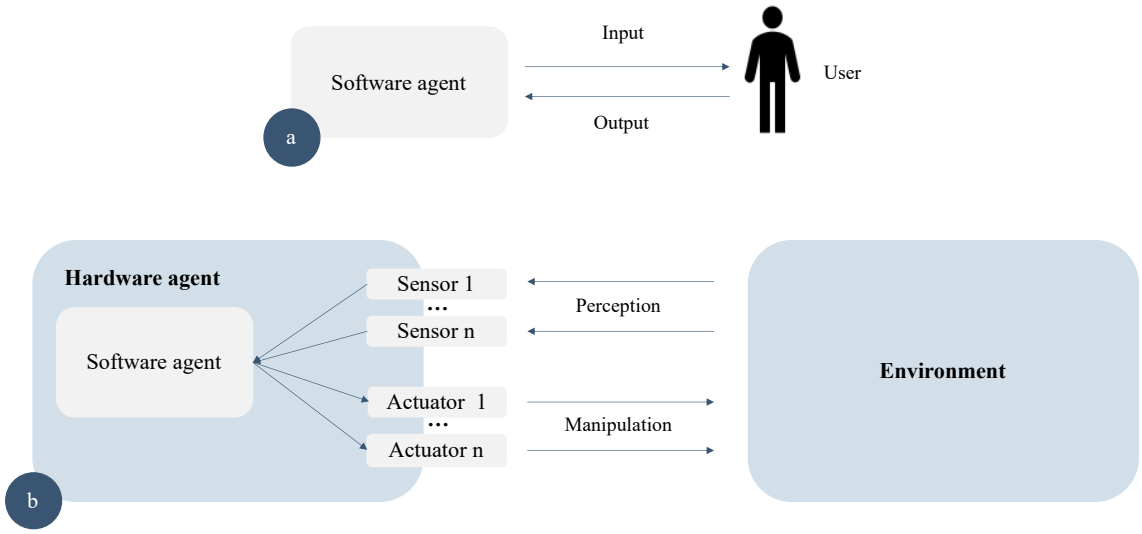

Figure 22. a) Software agent. b) Hardware agent (adapted from Ertel, 2017).

5.2.2 Artificial intelligence, machine learning, and deep learning gents

To understand the connection between AI, ML, and deep learning (DL), IBM Cloud Education (2020) contextualize them as follows:

- **Artificial intelligence:** The entire universe of computing technology that shows anything that resembles human intelligence. AI systems include anything from an expert system—system that can "think", solves problems, and makes decisions based on complex rules or logic—to something like a computer (or machine) that develops the intelligence, free-will, and emotions of a human being.
- **Manchine learning:** The subset of AI application that learns by itself. It reprograms itself as it considers more data, and it performs better (i.e., more accurate) the specific task it is designed for.

- **Deep learning:** The subset of ML application that teaches itself to perform a specific task with continuously greater accuracy, with no human intervention.

A special treat of human intelligence is adaptivity. As humans, we can adjust to different environmental conditions and shift our behavior accordingly through learning. Since our learning ability is more significant than that of computers, ML is a fundamental subfield of AI (Ertel, 2018). Figure 23 shows the main subsets that compose AI, where AI is the overall category that includes ML and natural language processing (Hurwitz & Kirsch, 2018).

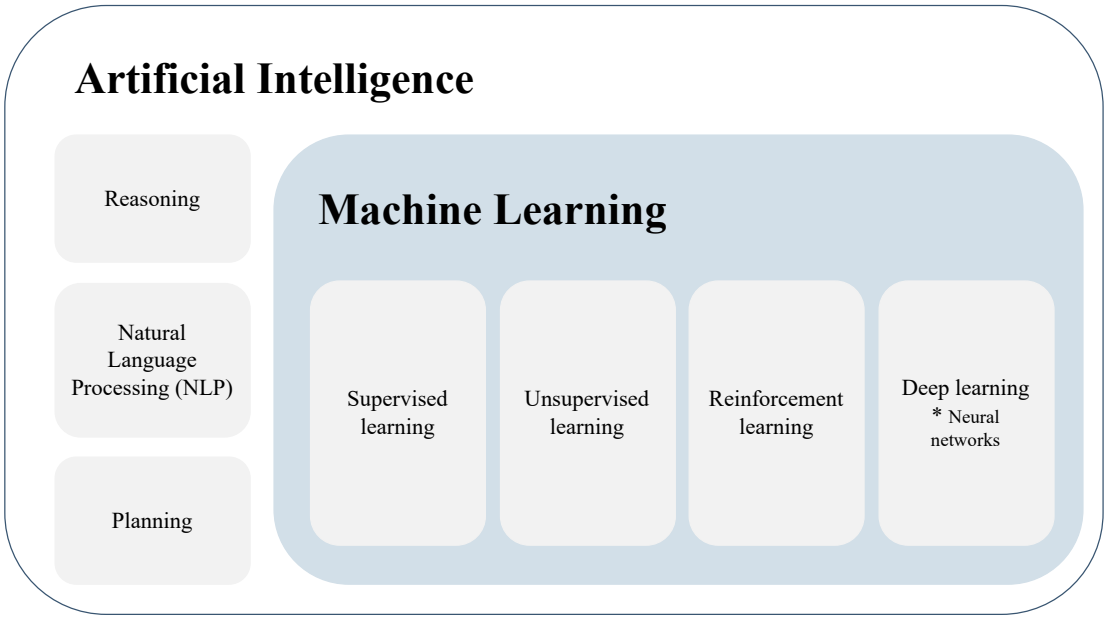

Figure 23. Categories inside AI (adapted from Hurwitz & Kirsch, 2018).

5.2.3 Machine learning

In line with the information reported by Hurwitz & Kirsch (2018), ML is a type of AI that allows a system to learn from data instead of explicit programming. Nevertheless, ML is not an easy process. ML uses several algorithms that iteratively learn from data to enhance/describe data and predict outcomes. As the algorithms take in training data, it is likely to have more accurate models based on the initial data set. A ML application is a result of the training process of a ML algorithm. After training, it is possible to get an output using an input and the obtained ML application. For instance, a predictive algorithm will create a predictive model. Then, it is possible to predict when using the predictive model and input data.

As stated by IBM Cloud Education (2020), ML applications are based on a NN, a network of algorithmic calculations that attempts to mimic the human brain's perception and thought process. In the most basic case, a NN consists of the following:

- *An input level*, where data enters the network.
- At least *one hidden level*, where ML algorithms process the inputs and apply weights, biases, and thresholds.
- An *output layer*, where various conclusions emerge.

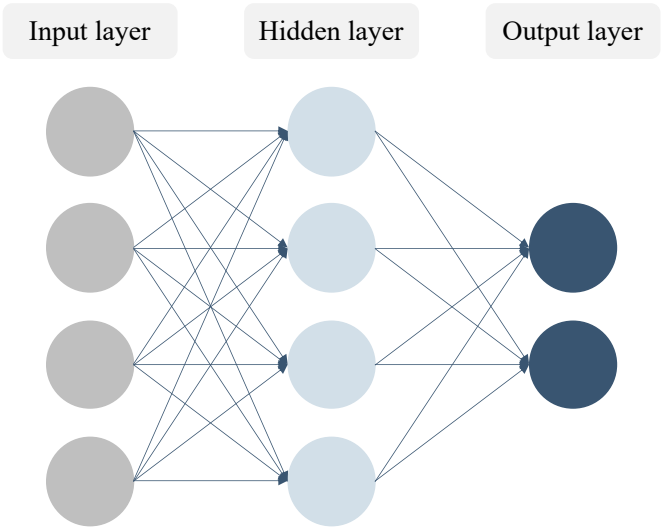

Figure 24. Basic NN (adapted from IBM Cloud Education, 2020).

ML techniques are required to improve the accuracy of predictive models. Depending on the problem's nature to address, there are different approaches based on the data type and amount.

- **Supervised learning:** Hurwitz & Kirsch (2018) stated that supervised learning typically starts with a set of data and some understanding of how that data is classified. Supervised learning finds patterns in data that is used for the analytics process, and this data has labelled features that define their meaning. For instance, if there are one million pictures of animals, including an explanation about each animal is considered a label. Then, it possible to create a ML application for distinguishing one animal from another. Besides, it makes a way to create categories related to the type of species. Since there is an identification of attributes and meaning of data, the user can understand the training and how it fits the labels'

details. If the label is continuous, it is a regression. Otherwise, If the label is discrete, it is a classification. Regression used for supervised learning helps to understand the correlation between variables—for example, weather forecasting. Through regression analysis, weather forecasting considers historical weather patterns and the current conditions to give a forecast on the climate. Supervised training models have broad applicability to various business problems, including fraud detection, recommendation solutions, speech recognition, or risk analysis.

- **Unsupervised learning:** Regarding unsupervised learning, Hurwitz & Kirsch (2018) sustained that is best suited when the problem requires a massive amount of data that is unlabeled. Understanding this data's meaning requires algorithms capable of understanding them based on classification, pattern, or clusters it finds. Unsupervised learning algorithms classify data in sets or clusters that share features. The unlabeled data creates the parameter values and classification of the data. The objective of this process is to add labels to the data so that the problem becomes supervised. Unsupervised learning can determine the outcome when there is a massive amount of data. So, unsupervised learning can be used as the first step before passing the data to a supervised learning process. Through unsupervised learning algorithms, companies can understand their data better. As supervised learning, unsupervised learning algorithms look for patterns in the data. However, the difference is that the data has not been already understood. In healthcare, for example, collecting data about a disease can help physicians obtain insights into the

symptoms' patterns and relate them to patients' outcomes. It would take a lot of time to label all the data sources correlated with a disease. Consequently, an unsupervised learning approach can help determine outcomes quicker than a supervised learning method.

- **Reinforcement learning:** For RL, Hurwitz & Kirsch (2018) affirmed that is a behavioral learning model. Since one of the aims of ML is to produce intelligent application systems, called agents, RL represents one approach to reach that goal. In an RL situation, the agent interacts with its initially unknown environment, examines the results of its actions, and adjusts its behavior suitably. The behavior described can be compared to the way biological beings learn.

Solutions based on ML can solve different types of tasks. Kang, Catal, & Tekinerdogan (2020) classify them as: i) regression, ii) classification, iii) clustering, iv) data reduction, and v) anomaly detection.

Regression

Regression maps the input features to a continuous variable. In order to obtain a minimum error in the prediction, ML algorithms optimize the coefficients of the independent variables. The resulting variable can take the form of an integer or floating-point number.

Classification

Classification maps input characteristics to one discrete output variable. The resulting variable outlines a class for the underlying problem. When the classification is binary, the resulting variable can take values of zero or one. When the classification is multi-class, the resulting variable can integrate several classes.

Clustering

Clustering segments data points into groups. The grouping process considers similar patterns between data points. Therefore, similar points are aggregated in the same group, so it is possible to extract relevant information from them.

Data reduction

The data reduction applications aim to decrease the number of characteristics in a data set. In this way, it is possible to identify and eliminate repetitive data points or those related to noise. Besides, it is possible to eliminate some positively correlated or not relevant features in the dataset for obtaining models faster. Data reduction is usually used as an auxiliary method for other ML applications, such as regression and classification.

Anomaly Detection

The purpose of anomaly detection is to group samples to find outliers. Usually, anomaly detection algorithms are handled with unsupervised learning methods.

5.2.4 Reinforcement learning

Figure 25 shows the basic design of RL system with its learning-action loops. In this type of system, an agent interacts with its environment to select the best possible action given a determined state (s_t), where the environment is in at step t. The action chosen by the agent changes the environment's state from s_t to s_{t+1}, and it generates a reward r_t for the agent.

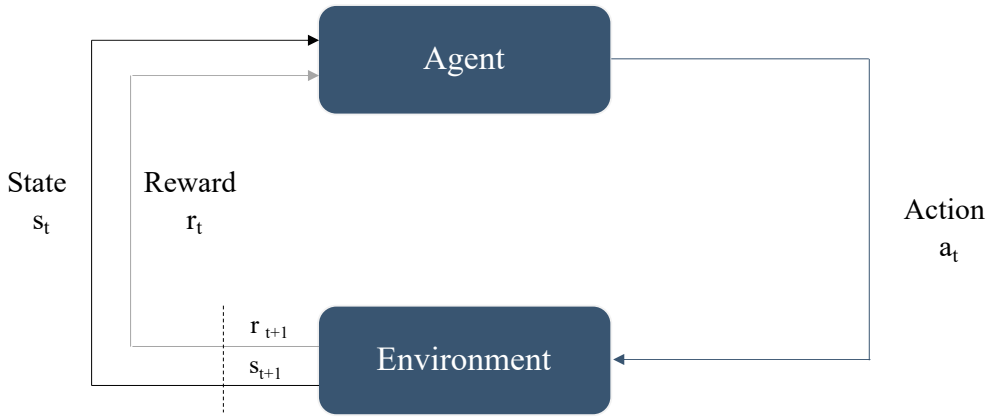

Figure 25. Design for a RL system (adapted from Sewak, 2019).

Later, the agent selects the best possible action for the new state (s_{t+1}), whereby it gets a reward r_{t+1}. Throughout a defined number of experiments for training the agent (iterations), the agent tries to improve its decisions when selecting the 'best action' that it can take in a given environment's state. To do so, the agent looks at the rewards that it received in the training process. The environment's role here is to display the agent with different possible states the problem might have (or a representative set), and that the agent need to react to. Besides, to guide the agents' learning process, the environment gives the

agent a reward or penalty that corresponds to the agent's action in a given state. The agent's goal is to minimize the expected long-term cost, considering all the time steps. Therefore, the reward is a function of the action and the state. This means that, for the same action, it is possible to have different rewards given different states (Sewak, 2019; Gabel & Riedmiller, 2008).

The agent in reinforcement learning

As stated by Sewak (2019), the agent is the most important part of an RL method since it holds the capacity to make decisions and suggest the optimal action for a given situation. In order to train the intelligence of the agent, it is necessary to create a representation of a similar environment that the agent will face. Besides, it is also needed to abstract the environment and the agent's context in the form of appropriate states. Given the agent's importance, it is essential to define the features related to the agent as follows.

The value function

According to Sewak (2019), the agent requires to choose which is the best action it can take when facing a specific state. In this way, the agent must identify the next best state to be in (possible to reach from the current state), determined by the history of present and future rewards that the agent received in that specific state earlier. Basically, with this technique, the aim is to predict the 'Value' (or utility) of 'seen' and 'unseen' states (or state-action combination), based on the 'seen' states. The ' (State) Value Function' is expressed

as V(s). This function summarizes the various problems in dealing with future/delayed and probabilistic/uncertain rewards by converting them into a homogenous function attached to a state that the agent will learn. The agent's decision is determined by identifying the most profitable state, considering the reachable possible states from the current state through a particular action (the recommended action). Later, the agent will recommend one action to execute a transition to the most gainful state from which the agent will have to choose the most lucrative state from this new state again. The transition stops when the agent has seen all the possible states. The agent's training tries to learn this 'Value Function', which is the states' value obtained through the training data.

The reward

Sutton & Barto (2018) reported that a reward signal defines the goal of a RL problem. On each time step, the environment sends a single number called 'reward' to the RL agent. The agent's objective is to maximize the total reward it receives over the long run. Thus, the reward signal defines what the good and bad events for the agent are. Rewards are the immediate and defining characteristics of the agent's problem, so they are the primary basis for changing the policy. When an agent selects an action following a determined policy and results in a low reward, it is necessary to change the policy to select other action for the same future situation. In general, reward signals may be stochastic functions of the environment's state and the actions taken.

The action-value (Q-function)

The definition of the 'Value Function' given by Sewak (2019) represents an indirect and general way to decide the best action. As a complement, the 'Action-Value Function' definition is a convenient way to consider the action as part of the 'Value Function'. This new definition is denoted by Q(s, a), also called 'Q-Function'. The 'Value Function' is just a function of the state (V(s)), while the 'Q-Function' depends on the state (s), and the action (a).

Exploring versus Exploiting

For training the 'Action-Value Function', it is possible to start with a function and initial random values or a fixed-heuristic initialization. The 'Action-Value Function' values are updated by executing experiments on the data/scenarios/episodes. If the agent gets inclined to any default value or an intermediate value from the 'Action/ Value Function'. In that case, it can result in the 'exploitation' of the learned information, and there will not be room for further improvement by 'exploring' new information. So, it is necessary to build a way where the agent can choose the decision criteria. In this way, two approaches can help the agent decide when to exploit or explore: i) On-Policy, and ii) Off-Policy (Sewak, 2019).

Policy and the on-policy and off-policy approaches

The 'Policy' (π) is the agent's strategy for taking the next best action based on the current state. It is defined as a function of the state *π(s)*, and it returns the best action to take for a state transition. The 'Policy' lasts valid for the whole learning/training phase (Sewak, 2019).

As defined by Ertel (2017), the objective is that the agent learns an optimal policy based on its experiences. There is an optimal policy if it maximizes reward in the long run (considering the observed process's time horizon). For defining the concept of 'maximize reward', it is necessary to define the concept of value of a policy p when we start in starting state s_t as:

$$V^{\pi}(s_k) = r_k + \gamma r_{k+1} + \gamma^2 r_{k+2} + \cdots = \sum_{i=0}^{\infty} \gamma^i r_{k+i} \qquad (4)$$

In equation 4, γ is a constant value between 0 and 1, which ensures a discount in future feedback. Certainly, the immediate reward r_t is weighted the strongest. Besides, the farther the feedback, the higher the discount made. A policy $p^{\star}$ is denominated optimal when for all states s:

$$V^{\pi^{\star}}(s) \geq V^{\pi}(s) \qquad (5)$$

This means that the policy $p^\star$ it is at least as good as all other policies according to the defined value function.

On-policy learning

The 'On-Policy' learning is the approach where the action with the highest probability of reward also has the highest probability of being selected. In this case, the 'Q-Function' is learned in a way that the probabilities -scaled to sum up to one- for each action are computed and choosing one action is stochastically based on the respective scaled probabilities of different actions. In this way, selecting a 'Policy' has the probabilistic nature of exploration and exploitation built in this approach (Sewak, 2019).

Off-policy learning

The 'Off-Policy' learning is the approach where exploitation is not in-built in the 'Estimation' policy. Therefore, a separate 'Policy' is required. In this learning mechanism, a defined probability (ε) determines the 'Exploration' of new actions rather than taking the predicted best action in the current state. When there is no 'Exploration' (with probability $1-\varepsilon$), there will be 'Exploitation' (known as 'Greedy' behavior), and the action is taken based on the best 'Q-Value'. Generally, for 'Off-Policy' approach, the policy for estimation/update of the 'Q-Function', and a different policy for the 'Behavior' of the agent (Sewak, 2019).

5.2.5 Actor-critic in reinforcement learning

In the context of RL, Grondman, Busoniu, Lopes & Babuska (2012) explained that algorithms can be classified into three groups according to the definition of the policy (actor) and value function (critic): i) actor-only, ii) critic-only, and iii) actor-critic.

Actor-only methods and the policy gradient

The actor-only approach usually uses a parameterized group of policies, so optimization procedures over them are used directly. The benefit of using a parameterized policy is the generation of a continuous spectrum of actions, but the optimization methods used (usually policy gradient methods) show high variance for the gradient estimation. In this way, it leads to a slow learning process. Policy gradient methods are mainly actor-only and do not consider any form of a stored value function. Alternately, most actor-only algorithms work with a parameterized family of policies and optimize the cost defined by the equation 6 over the parameter space of the policy:

$$J(\pi) = E\{\sum_k^{\infty} \gamma^k r_{k+1} \,|d_0, \pi\}$$

$$= \int_X d_\gamma^\pi(x) \int_U \pi(x,u) \int_X f(x,u,x')\, \rho(x,u,x') dx' du dx \qquad (6)$$

Where $J(\pi)$ is the cost function, that is equal to the expected value of the discounted sum of rewards (discounted return) when starting from an initial state $x_0 \in X$ drawn from an initial state distribution $x_0 \sim d_0(\cdot)$. In the Equation 6:

- $d_{\gamma}^{\pi}(x) = \sum_{k=0}^{\infty} \gamma^k p(x_k = x | d_0, \pi)$ is the discounted state distribution under the policy π.
- γ is the discount factor between one and zero.
- $p(x_k = x)$ is a probability density function.

Compared to critic-only methods, actor-only methods allow the policy to generate actions in the entire continuous action space.

A policy gradient method is generally obtained by parameterizing the policy π by the parameter vector $\vartheta \in \mathbb{R}^p$. Since the cost function (equation 6) depends on a parameterized policy p_{ϑ}, it is a function of ϑ. Supposing that the parameterization is differentiable by ϑ, the gradient of the cost function with respect to ϑ is defined by:

$$\nabla_{\vartheta} J = \frac{\partial J}{\partial \pi_{\vartheta}} \frac{\partial \pi_{\vartheta}}{\partial \vartheta} \qquad (7)$$

Then, it is possible to find a locally optimal solution for the cost J through standard optimization methods. In this way, the gradient $\nabla_{\vartheta} J$ is calculated for each time step, and then the parameters are updated in the gradient's direction. For instance, a simple gradient ascent method would yield the policy gradient update equation:

$$\vartheta_{k+1} = \vartheta_k + \alpha_{a,k} \nabla_{\vartheta} J_k \qquad (8)$$

Where $\alpha_{a,k} > 0$ is a small enough learning rate for the actor, which it allows to obtain that $J(\vartheta_{k+1}) \geq J(\vartheta_k)$.

An advantage from actor-only approaches is their convergence property, a property usually brought by the gradient descent methods. Hence, it is possible to obtain convergence if the calculated gradients are unbiased and the learning rates $\alpha_{a,k}$ satisfy:

$$\sum_{k=0}^{\infty} \alpha_{a,k} = \infty \qquad \text{and} \qquad \sum_{k=0}^{\infty} {\alpha_{a,k}}^2 < \infty \qquad (9)$$

A disadvantage in this approach is that the determined gradient may have a high variance. Besides, without knowledge of previous estimates, it is possible to calculate each gradient.

Critic-only methods

The critic-only method uses temporal difference (TD) learning and presents a lower variance for estimating predicted returns. A direct way to obtain a policy in critic-only methods is by selecting greedy actions, in which the expected return is the highest according to the value function. However, it is necessary an optimization procedure in every state to find the action that leads to an optimal value. When the action space is continuous, this approach can be computationally intensive. Hence, critic-only methods usually discretize the continuous action space, and then execute a faster optimization

procedure. It is expected that this approach undermines the capacity to use continuous actions and find a true optimum.

Critic-only approaches, such as Q-learning and SARSA, utilize a state-action value function, whereas there is no explicit policy function. When the state and action spaces are continuous, the state-action value function is an approximation. These methods find online an estimated solution for the Bellman equation (equation 10) to learn the optimal value function (equation 11):

$$Q^{\pi}(x,u) = E\{\rho(x,u,x') + \gamma\, Q^{\pi}(x',u')\} \qquad (10)$$

$$Q^{*}(x,u) = E\{\rho(x,u,x') + \gamma\, max_{u'} Q^{*}(x',u')\} \qquad (11)$$

Where $Q^{*}(x,u)$ is the optimal state-action value, ρ is the reward function, x is the current state, u is a current action, x' is the resulting state, g is the discount rate, and u' is an action in the resulting state.

A deterministic policy, expressed as π: $X \rightarrow U$ (where X denotes the state space, and U denotes the action space), is calculated using an optimization procedure over the value function:

$$\pi(x) = \arg max_{u}\, Q(x,u) \qquad (12)$$

There is no reliable guarantee on the resulting policy's near-optimality for just any approximated value function when learning in an online setting.

Actor-critic algorithms

In this thesis, actor-critic method was explored for solving the scheduling problem since it combines the advantages of actor-only and critic-only methods. As actor-only methods, actor-critic approaches can provide continuous actions, while a critic tackles the high policy gradient variance given by actor-only methods. In these types of approaches, the critic evaluates the current policy prescribed by the actor, and the actor approximates and updates the value function using samples. Then, the value function is used to update the actor's policy parameters to improve the performance. Figure 26 shows the conceptual design of this method.

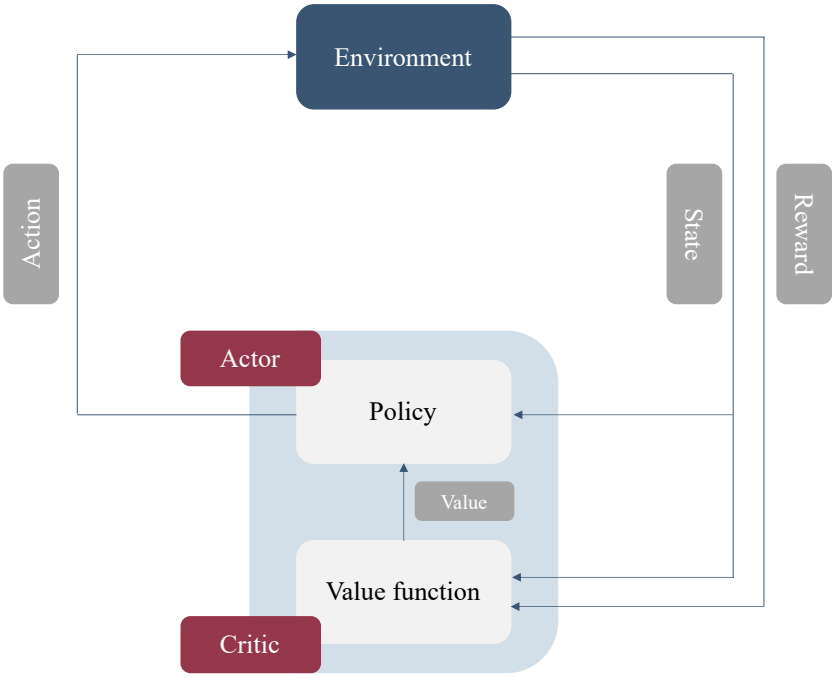

Figure 26. Conceptual design of the actor-critic method (adapted from Sewak, 2019).

The objective of actor-critic algorithms is to find the best policy possible. A requirement to do this is the capability of the critic to evaluate a given policy. Namely, the critic must find an approximate solution to the Bellman equation (equation 10) for that policy. The difference between the right-hand and left-hand sides of the Bellman equation is called the TD error, and it is utilized to update the critic. Using the function approximation for the critic and a transition sample $(x_k, u_k, r_{k+1}, x_{k+1})$, the TD error is calculated as:

$$\delta_k = r_{k+1} + \gamma V_{\theta_k}(x_{k+1}) - V_{\theta_k}(x_k) \qquad (13)$$

Where V_{θ_k} is the value function parameterized by the vector q. If the parameterization is linear, the features (basis functions) will be denoted with f as:

$$V_\theta(x) = \theta^T \phi(x, u) \qquad (14)$$

For the convergence for actor-critic algorithms, the critic's estimate must be at least asymptotically exact. In this manner, the step sizes $\alpha_{a,k}$ and $\alpha_{c,k}$ must be deterministic, nonincreasing, and satisfy:

$$\sum_k \alpha_{a,k} = \infty, \qquad \sum_k \alpha_{c,k} = \infty \qquad (15)$$

$$\sum_{k} {\alpha_{a,k}}^{2} < \infty, \qquad \sum_{k} {\alpha_{c,k}}^{2} < \infty, \qquad \sum_{k} \left(\frac{\alpha_{a,k}}{\alpha_{c,k}}\right)^{d} < \infty \qquad (16)$$

For some d ≥ 0. The learning rate $\alpha_{J,k}$ is usually set equal to $\alpha_{c,k}$. The assumption made over the learning rate are typically set for all RL algorithm. They guarantee that learning will slow down but never stops. At the same time, they guarantee that the actor's update operates on a slower timescale than the critic to ensure that the critic has enough time to evaluate the current policy.

Chapter 6 - Application Development

This chapter presents the technical architecture for the model developed using a reinforcement learning method. It presents the building blocks for the developed model and its corresponding functionality according to Sewak (2019), the architecture for its implementation, the architecture of the final model, the production system definition, criteria for scheduling technique, and the actor-critic approach in Python.

6.1 Construction of the environment

The purpose of the environment is to present the state and provide relevant responses to train the agent. The environment used in this thesis is made of two primary functions: the 'step' and 'reset' functions. These functions are essential for the proper functioning of the environment's purpose. Besides, the environment also has additional functions to increase the applicability of the environment to different purposes. Following, we show the corresponding explanation for each function, their inputs, and their outputs.

6.1.1 The step method

In RL, the training is made through a mechanism that gives a state to the RL agent. Then, the agent takes the best possible action for that given state. Hence, it is necessary a mechanism that provides to the agent a reward (or penalty) corresponding to the chosen action. Besides, the same mechanisms must return the subsequent change of state related

to the action. The new state is then served again to the agent for taking the following action and so on, as shown previously in the figure 25.

The responses that the environment delivers are the new states and rewards associated with the action taken (from each current state). In this way, the environment needs a method, the 'step' function, that receives an action and returns a state (observation), reward, Boolean variable ('done'), and other descriptive variables. A description of these set of elements is as follows:

- **State (observation):** This variable contains the state (current/next) given by the environment when the agent takes one action, which is the input for the 'step' method. This state has observations that represent in the best possible way the environment.
- **Reward:** It is the agent's instantaneous reward for reaching a new state after taking one action (input). This represents the instantaneous reward (for one 'step'). If the accumulated reward is necessary, it must be maintained outside this method.
- **Boolean variable (done):** This is a Boolean flag which deals with episodes. An episode is one experiment with a beginning and an end. The problem's nature (e.g., number of machines, number of tasks, constraints, etc.), type of algorithm, and computational cost determine the number of episodes. When one episode starts, the variable 'done' starts as 'False', and when the episode finishes, the variable takes the value of 'True'. A new episode starts when the environment is 'reset', which is discussed further in this chapter. If the environment is in 'done' state (i.e., done ==

True), the step function does not work since taking one next step is not feasible for that episode. So, the episode ends.

6.1.2 The reset method

The reset function takes no arguments and returns variables and a state corresponding to the start of a fresh episode. When the environment is initiated for the first time, or when a new episode starts, the environment's state and its variables are reset through this function.

6.2 Architectural consideration for deep learning

The type and number of layers in a DL network depend upon the DL network's specific use. In this sub-section, we show the different considerations when working with DL.

6.2.1 Activation functions

In NNs, neurons need an activation function to transform the weighted input into an output. A change in the activation function leads to DL success and helps NNs to scale. In this manner, the activation functions used in this thesis are explained as follows:

- **Softmax activation:** Considering that the number of classes in the data corresponds to the numbers of neurons in the output layer (for classification application), the

SoftMax Activation layer gives the class probabilities for each class under consideration, scaled to the sum of all the class' class probability.

$$P(y = c|x) = \frac{e^{x^T w_j}}{\sum_{k=1}^{K} e^{x^T w_k}} \qquad (17)$$

Where:

k=1, 2,…,j,…,K is the number of classes.

$x^T w_j$: input vector.

y: output vector.

- **Linear and identity activation:** For regression problems, the output layer may have a single neuron and require as output from it a value corresponding to the variable's estimate in the output. Often this value is a scaled representation of the data in hand instead of an absolute representation of it. For RL, the Value Function estimation is an example of where linear activation is used in the output layer.

$$Identity\ (x) = \sum_i x_i\, w_i \qquad (18)$$

Where:

x_i: input that are provided to the neuron

w_i: weight of each input

- **Rectifier linear unit (ReLU):** ReLU is an activation function for the input and hidden layers that solves the vanishing gradient problem and inactive or dead neurons' problems. Equations 19 shows the activation function, and equation 20 shows a variant called Leaky ReLU.

$$ReLU(x) = max(0, x) \qquad (19)$$

$$Leaky\ ReLU(x) = \begin{cases} ax\ if\ x < 0 \\ x\ if\ x \geq\ 0 \end{cases} \qquad (20)$$

6.2.2 Loss function

It is needed to backpropagate the error to optimize the weights/hyperparameters in each layer for training a network. This loss/error is computed using a loss function, which takes the actual and predicted values to output the loss. There are two categories for loss functions: i) L1 loss, based on the absolute difference between the actual and predicted value, and ii) L2 loss, based on the square of the difference between the actual and predicted values.

6.2.3 Optimizer

In DL, the loss function's objective is to pick the ideal combination of different hyperparameters that minimize the loss. It is possible by finding the gradient of the loss function. If the loss function is convex (it is possible to find a minimum), updating the weights towards the loss function's gradient will result in a minimum value (local or

global). However, this task is not simple due to the high cardinality of the parameter space, the number of hyperparameters, and the complexity of the function to optimize. There are various obstacles to overcome while training in the gradient hyperspace, such as stagnation in a local minimum, gaps, and saddles. In this way, a good optimizer must be efficient, fast, and robust against these obstacles. In this thesis, we used ADAM (adaptative moment estimation), as discussed by Sewak (2019), compared by Riemer-Sørensen & Rosenlund (2020), and presented by Kingma & Ba (2014).

6.3 Convolutional neural networks

A convolutional neural network (CNN) means that the network uses a mathematical operation called convolution. Convolutional networks are a specific type of NN that uses convolution instead of general matrix multiplication in at least one of their layers (Goodfellow, Bengio & Courville, 2016).

A CNN consists of an input layer, hidden layers, and an output layer. For a feed-forward network, the middle layers are called hidden because of the masking of their inputs and outputs by the activation function and final convolution. In a CNN, there are hidden layers that perform convolutions. It includes a layer that does multiplication or other dot product, and its activation function is commonly ReLU (Albawi, Mohammed & Al-Zawi, 2017).

Convolutional layers

The term convolutional in CNN regards using a convolutional layer, which is named due to the convolution function. A convolution is a mathematical operation on two functions resulting in a third one, the altered (convoluted) version of one of the original functions. Convolutional layers use cross-correlations that are very similar to the convolution operation. Each convolution layer's output is a set of objects called feature maps; each map generated by a single kernel filter. Then, the feature maps define a new input to the next layer. Each convolutional layer could have a different number of kernel filters. For a CNN, the input is a fourth-dimension tensor shape. After passing through a convolutional layer, the input becomes abstracted to a feature map, with a fourth-dimension shape.

6.4 Architecture for actor-critic implementation

For taking one action in reinforcement learning, a policy is necessary. In actor-critic, for permitting the actor to take an action, it is needed to provide to the actor a policy. Then, the actor has a stochastic policy that utilizes to take an action in every step of the training process. At the same time, the actor improves and updates its policy through the policy-gradient approach. The policy is updated continuously using the state value estimation received from the critic. Figure 27 shows the architecture for the implementation of actor-critic method.

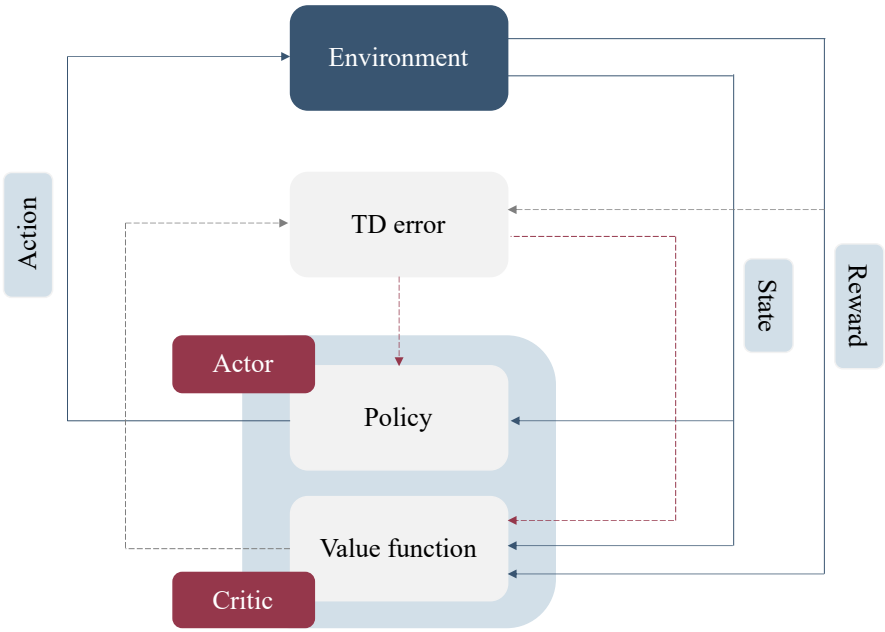

Figure 27. Actor-critic implementation

The error between the estimation of the value for the subsequent states is calculated using equation 21, which uses the instantaneous reward and discounted state value of the subsequent state:

$$\delta_t = r_t + \gamma V(s_{t+1}; w_t) - V(s_t; w_t) \qquad (21)$$

where, γ is the discounting factor. Besides, equation 21 corresponds to one-step return updates, similar to TD (0) without the eligibility traces. When needed, it is possible to expand this equation to an n-step update. V is the state-value estimator function parametrized by weight/parameter vector w. With every step, the value estimator function's

weight vector also needs to change at every step. In this way, at the t_{th} step, this parameter vector could be denoted as w_t. The value for any terminal state is set as 0.

The error at time t is estimated using equation 21. In this manner, the value estimator function of the critic is updated for each subsequent step. Due to the weight vector w parameterizes the value estimator function, the update of the weight vector w from w_t to w_{t+1} for each iteration updates the value estimator function. So:

$$w_{t+1} = w_t + \alpha_w \delta_t \Delta_w V(s; w_t) \qquad (22)$$

where, α_w represents the learning rate for the value estimator function.

Like the update of value estimator function for the critic, the actor's policy estimator function updates each iteration through equation 23:

$$\theta_{t+1} = \theta_t + \alpha_\theta \delta_t \Delta_\theta log\pi(a|s; \theta_t) \qquad (23)$$

where π is the policy estimator function, which is parameterized over the vector θ. The vector θ also updates to θ_{t+1} from θ_t in each iteration, as shown in equation 23. For this case, α_θ is the learning rate for the update of the policy estimator function.

6.5 Contribution architecture

The solution presented in this thesis used actor-critic approach. Considering the information given in the previous subsection, the model has three main functional modules: i) the trainer, ii) the actor-critic model, and iii) the environment. The trainer is the functional module where all the episodes and all the steps are executed. It chooses an action, given by the model, that is sent to the environment of the problem. The environment, that needs as inputs the process time matrix, a sample of jobs from the list of work orders, and an action on each step, returns a state and an instantaneous reward depending on the chosen action provided by the model. And the model, that is the actor-critic algorithm. Then, the process is repeated depending on the number of steps, the number of episodes, or if the algorithm can or cannot find a solution. Figure 28 shows the architecture of the solution developed in this thesis.

6.6 Production system definition

In this thesis, we chose a parallel machine production system with intermittent arrivals because it is affected by scheduling problems, and the solution has not been explored enough through the RL approach. In this way, we set 6 machines and 12 job-cluster job-shop, with a list of 100 jobs from a pool of 500 jobs to be done (from different job clusters). Assumptions we considered over the system are provided below:

- The machines can process specific job clusters, not all of them. Therefore, there are three families of jobs and each of them can be done in two specific machines.

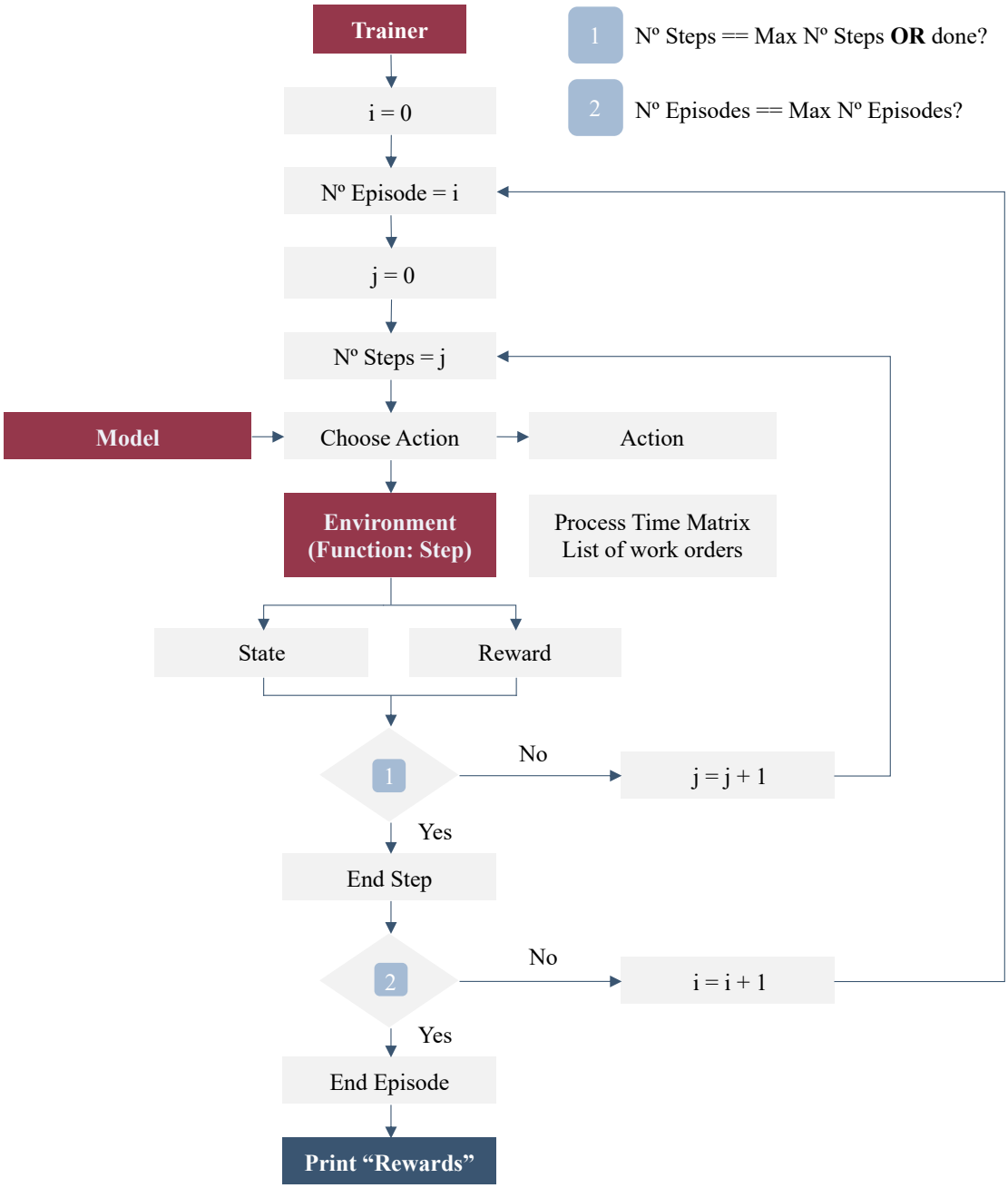

Figure 28. Flow diagram of the architecture for the solution developed.

- A machine can process just one job at the same time during and without being interrupted.

- Setups are included in the processing time of each machine-job combination.
- Resources are renewable.
- The job goes through just one machine and leaves the system.
- There are new arrivals among all the steps in an episode.
- Due dates are not considered.
- Perturbations and material handling times are not considered.

Notation

j = job (j = 1, 2,…,100)

i = machine (i = 1, 2,…,6)

Number of machines = 6

Number of job cluster = 12

Number of jobs = 100

Machine job = process time to produce job j of certain job_cluster in machine i.

Work order = list of jobs to be processed and its respective job cluster.

The objective function is to minimize the flow time and to maximize the machine utilization of the production system.

6.7 Criteria for scheduling technique

Assuming a stochastic behavior of the system by new jobs arrival and no constraints regarding jobs due dates, we chose process time-based dispatching rules that are highly related to the minimization of makespan, hence minimization of flow time and maximization of machine utilization in production systems (Holthaus & Rajendran, 1997). We used a hierarchy of dispatching rules, and in the first level, we applied SPT. For the second one, we applied FIFO to solve the scheduling problem.

6.8 Actor-critic approach in Python

We adopted *Python* version 3.8.5 in *conda* 4.9.2 to training DRL agents, in our case, the actor-critic agent. In Python, we used the libraries: i) Pandas, a library to use data structures and data analysis tool, ii) Numpy, a library to create n-dimensional arrays in the developed code. As pandas, it is accessible for programmers, and it allows mathematical operations, generation of random numbers, and linear algebraic functions, iii) Pytorch, a library created to lead with ML applications such as analysis of images and natural process languages, and it is based on *Torch* library, and iv) Torch, a package that consists of data structures for multi-dimensional tensors to enable mathematical operations.

6.8.1 Technical implementation

To create and train a NN, *Pytorch* contains a module called *Torch.nn*. It provides functionalities to create random or zero-filled tensors to be used as weights and biases for a linear model, adding a gradient. This gradient will help *Pytorch* record all the operations done on the tensor and calculate the gradient while backpropagation. This module contains the following building blocks that we used in this thesis:

- **Convolution Layers – nn.Conv2d:** Apply 2-Dimension convolution to an input signal that is comprised of many input planes. The class *torch.nn.Conv2d* is given by:

  torch.nn.Conv2d(in_channels, out_channels, kernel_size, stride=1, padding=0, dilation=1, groups=1, bias=True, padding_mode ='zeros')

  in_channels = num_inputs given by the state and hx, cx tensors.

  out_channels = 64

  kernel_size = 3

  Stride = 2

  Padding = 1

- **Long Short-Term Memory – nn.LSTMCell:** LSTM network is part of the category of Gated Recurrent Neural Networks (RNNs) and it is responsible to get accurate models for time series data. Those networks apply modules chain that are repetitive and four layers interacting among them. The LSTM network manages

information adding or removing it, on the cell state through switching the gates constructed by a sigmoid function and a point multiplier. The four layers or gates are: input, external-input, forget, and output gates. They do internal operations on each LSTM module and include an internal state to remember the values linked to a cell in a hidden state that are the module output. The class *LSTMCell* is given by:

torch.nn.LSTMCell(input_size, hidden_size, bias=True)

input_size = 400

hidden_size = 256

The *LSTM* is composed by inputs, hidden state (*hx*) and cell state (*cx*).

inputs = tensor composed by the features of the input sequence

hx = It is a tensor that contains the hidden state for each element in the batch.

cx = It is a tensor that contains the cell state for each element in the batch.

- **Neural Net Linear – nn.Linear:** It is used to apply a linear transformation to the incoming data. The class *Linear* is given by:

  torch.nn.Linear(in_features, out_features, bias=True)

  in_features = 256 (for critic and actor linear)

  out_features = 1 (for critic_linear) and num_output (for actor_linear), where the num_output is the action space, and it is a tensor composed by the action dimension. In this case it is a tensor of *6 x 1.*

  The *nn.Linear* class has two variables that we employed in the development of the code:

- ***Linear.weight:*** Composed by learneable weights of the module.
- ***Linear.bias:*** Composed by learneable bias of the module.

6.8.2 Functions and parameters definition

Regarding the actor-critic method, we defined the following functions and parameters in the environment and architecture:

Environment

- **State:** It is a torch vector composed of the *job status* and the *job distribution time*. The *job status* is a binary *numpy array* (1 for a job waiting to enter into the system, 0 for a job in the system). The *job distribution time* is also a *numpy array* that counts the waiting time of each job. The size of *job status* and *job distribute time* vector is defined by the number of jobs, which permits to characterize the final size of the state vector as the previously mention merge. The *state vector* needs to be converted into a *torch vector*, allowing its use in the actor-critic class.
- **Action:** The action is a *Torch vector* that depends on the *number of machines*. It is a binary vector (1 for an available machine, 0 for an unavailable machine). This binary connotation is related to the probability of doing an action with a good reward and exploring the different alternatives in the set space-state.

- **Step:** This function represents the environment from which the agent will learn how to obtain the highest possible reward given a certain initial state, taking a certain action. In this thesis, we defined the step function using the following parameters:
    - ***Job identifier:*** This is a vector, and its size depends on the *number of machines*. It has applied the dispatching rules explained before (SPT and FIFO) to choose each machine's most suitable jobs. This vector is updated on each step.
    - ***Job status:*** This is a vector, and its size is defined by the *number of jobs.* This vector shows the status of the jobs (0,1). It takes the value of 1 for a pending job, and 0 for a job chosen in a specific machine. Moreover, *job status* is part of the *state* vector.
    - ***Job distribution time:*** This is a vector that measures the waiting time of a job since it is assigned in the *job identifier* vector, but it is not chosen to go into the system. It will be updated if the job is not executed in previous steps.
    - ***Status of the machine:*** This is a vector, and its size depends on the *number of machines*. It is set to know if the machine is available or not to receive a new job. This vector updates its component values to 0 or 1 (1: machine is not available, 0: machine is available).
    - ***Job processed by a machine:*** This is a vector, and its size depends on the *number of machines.* This vector stores information about the current

identifier of the job that is being processed in a specific machine for the current step. When a job is finished, the vector goes through a transition state (empty) to wait the next job in the next step.

- ***Job processing time:*** This is a vector, and its size depends on the *number of machines*. It stores information about the current time units passed to produce a job *j* in machine *i*, measured in the current step. When a job is finished, the vector goes through a transition state (empty) to wait the next job in the next step.
- ***Processed jobs:*** List of jobs done among the steps. This list is not reset among steps.
- ***Used machines:*** List of machines that processed the jobs belonging to the *job processed* list. This list is not reset among steps.

- **Done:** The 'Done' function in this thesis is a Boolean expression based on the number of available jobs in the pre-shop pool. We characterized the number of available jobs as *left jobs.* Figure 30 shows the functionality of the 'Done' function.

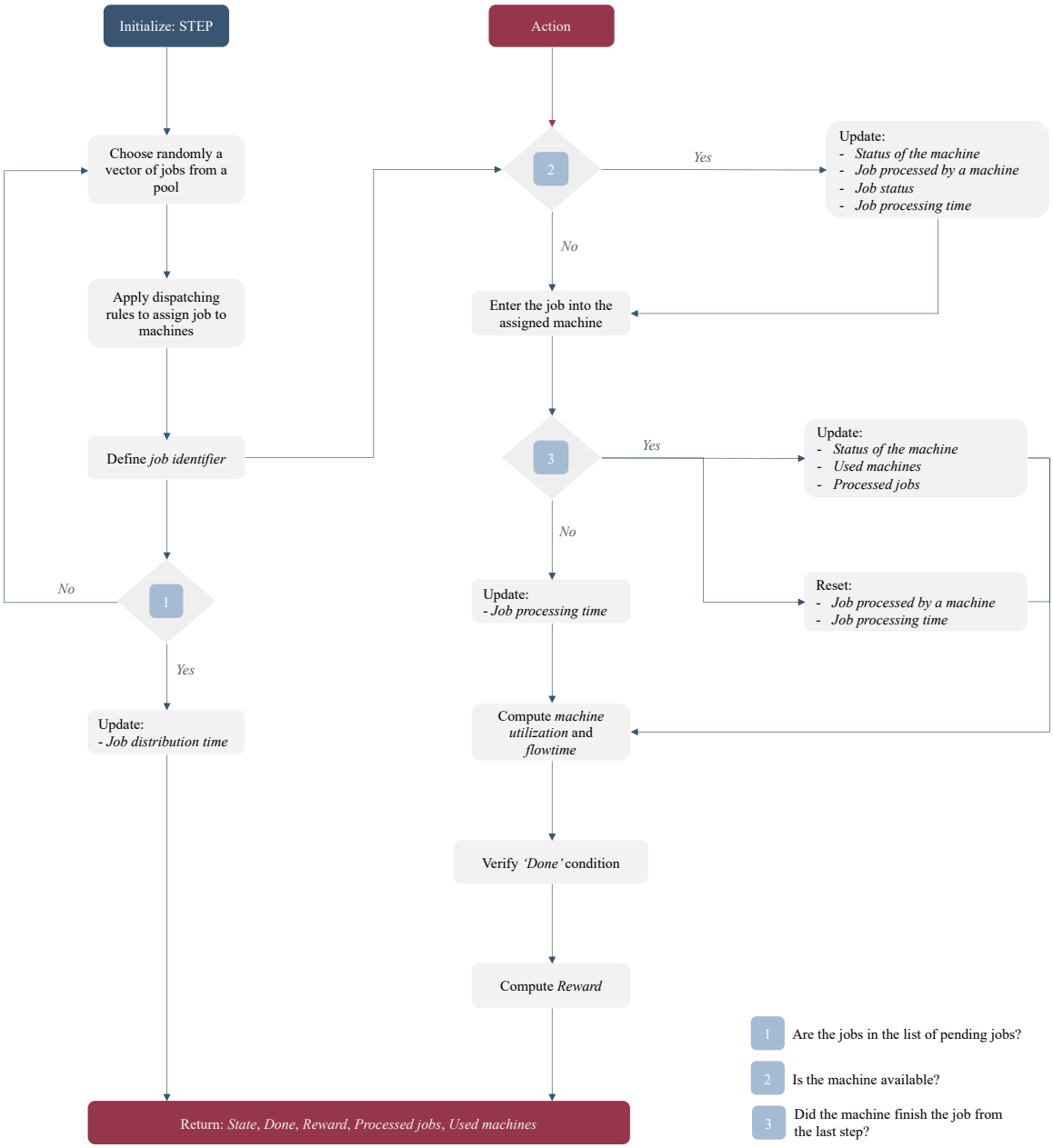

Figure 29. Flow diagram for Step function.

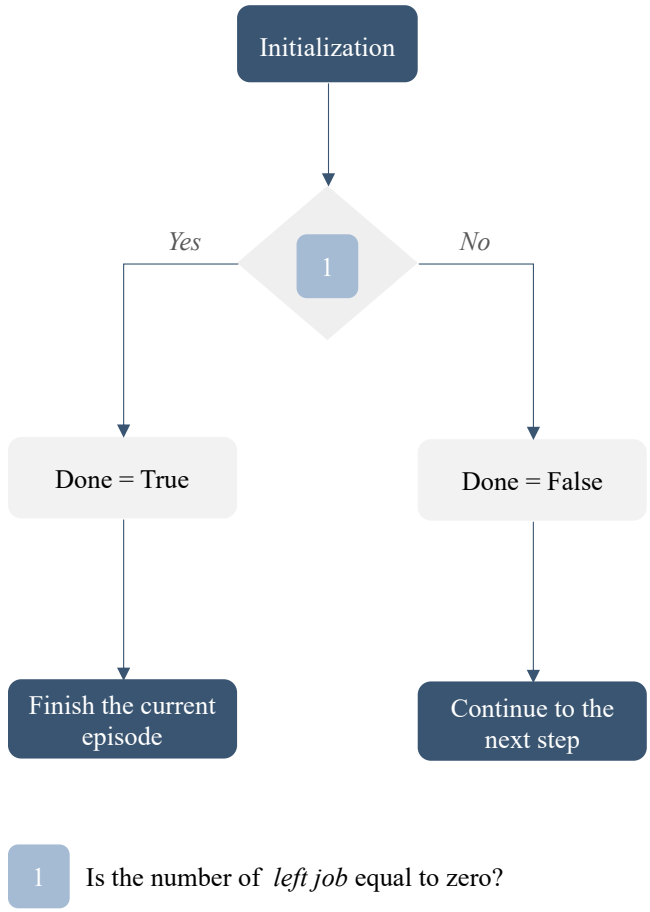

Figure 30. Flow diagram for Done function.

- **Reward:** The reward function represents the maximization of *machine utilization* among all the machines, and minimization of *flowtime* through a linear combination of those performance indicators.

$$
\begin{aligned}
&Total\ reward \\
&\qquad = 0.5\ (reward\ machine\ utilization) \\
&\qquad + 0.5\ (reward\ flowtime) \qquad (24)
\end{aligned}
$$

To compute it, we must define the following parameters:

- ***Flow time:*** This is a vector, and its size depends on the *number of machines*. It is based on the concept of "mean flow time", defined as the average time that a job spends on a production system. The formula is:

$$Mean\ Flow\ Time = \frac{\sum_{k=1}^{N}(N-k+1) * PT_k}{N} \qquad (25)$$

Where:

N: Total number of jobs done

k: job k

PT_k: Processing time of job k

- ***Machine utilization limit:*** This is a scalar value applied as a new machine status in case of transition between two jobs in a machine.

- ***New machine status:*** This is an internal vector that was created to consider the transition state between the completion of a job and the new job to be done in a specific machine. In this transition, we consider the *machine utilization limit* value.

- ***Average machine utilization:*** This is the average machine utilization among all the machines. It is computed as:

$$Average\ machine\ utilization = \frac{\sum_{o}^{i} n_ms[i]}{machines} \qquad (26)$$

The computation of the reward depends on two conditions: 1) the existence of jobs in the *job identifier* vector and, 2) the existence of processed jobs in the system. Figure 31 shows the process that the code follows for obtaining the reward.

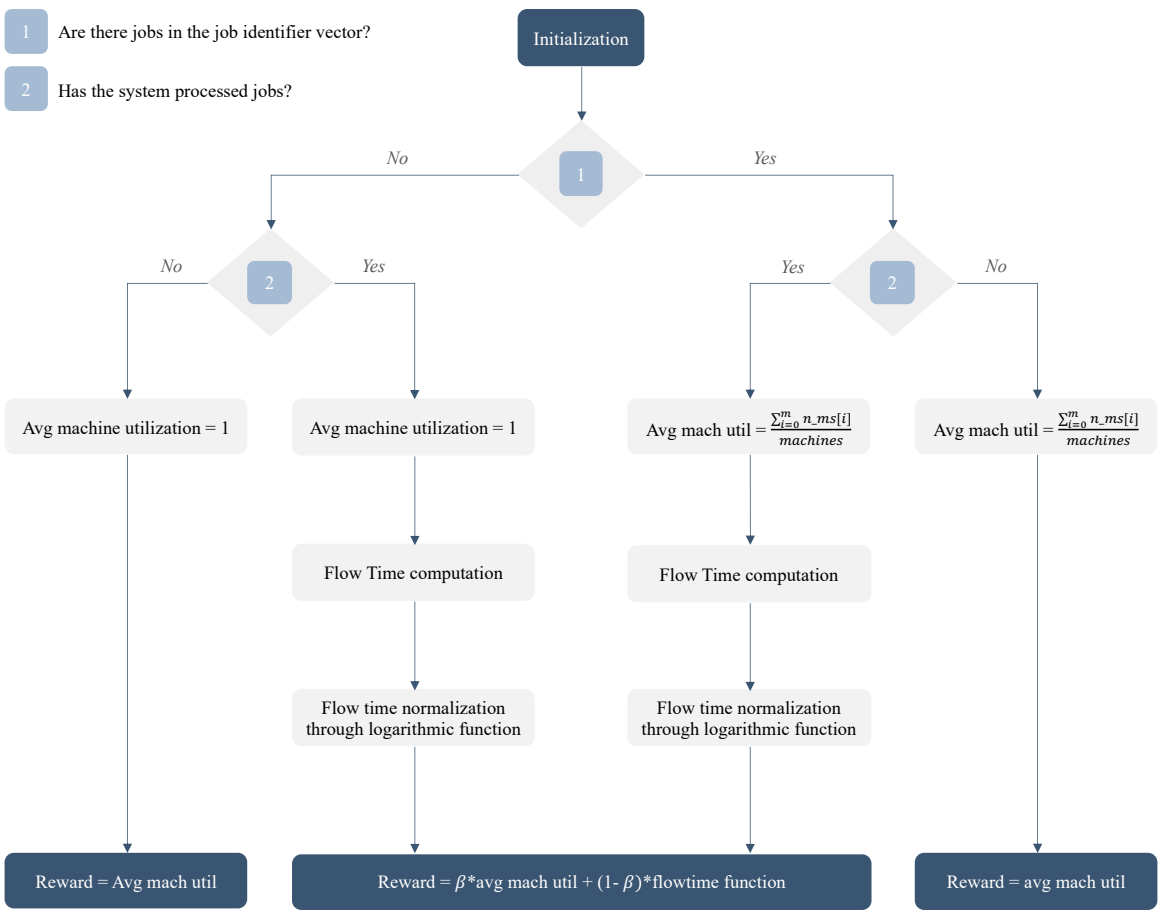

Figure 31. Flow diagram for reward computation.

Architecture

- **Activation function:** As explained in the previous chapter, the activation function is important for converting from a weighted input to an output in a neuron. It is possible to find functions to activate the input layers (as softmax and log_softmax) and to activate the output layers. In the torch.nn module, there are the following functions:

  Rectified Linear Unit - ReLU: torch.nn.functional.relu(input, inplace=False)

  input = Conv2d

  inplace = False

  Softmax: torch.nn.functional.softmax(input, dim=None, dtype=None)

  input = logit (or actor_linear)

  dim = -1

  Log_softmax: torch.nn.functional.log_softmax(input, dim=None, dtype=None)

  input = logit (or actor_linear)

  dim = -1

- **Loss function:** For this thesis we implemented Huber loss function, that is part of the L2 loss, based on the difference between the actual and predicted values. Huber loss deals better with outliers than a Mean Square Error (Mahendru, 2020).

$$L_\delta = \begin{cases} \frac{1}{2} * \left(y - f(x)\right)^2, if\ |y - f(x)| \leq \delta \\ \delta|y - f(x)| - \frac{1}{2} * \delta^2 \end{cases} \quad (27)$$

- **Optimizer:** We employed Adam optimizer because it is efficient for stochastic optimization, and it just requires first-order gradients with little memory requirement. This method calculates the adaptive learning rates of different parameters that start with estimations on the first and second moments of its gradients (Kingma & Ba, 2014).

6.9 Experiments

The model architecture discussed previously was used to address the scheduling needs of the chosen production system. In this way, we run a simulation using the parameters shown in table 6. The processing time matrix is shown in table 7. Once the simulation was run, we obtained profiles over episodes for the total reward, machine utilization, flow time, and the reward function associated with the flow time. Annex A contains the supporting code developed for running the experiments and obtaining results.

Table 6. Parameters for running the experiment.

Parameter	Experiment 1
Learning rate (α)	0,0001
Discount factor for rewards (γ)	0,99
Generalized Advantage Estimation (λ)	1
Entropy coefficient	0,01
Value loss coefficient	0,5
Gradient normalization	50
Random seed	1
Number of steps	300
Episode length	300
Episodes	5000
Error tolerance	0,001
Number of resources	6
Number of product types	12
Total jobs	500
Jobs per episode	100
Machine utilization factor (β)	0,5
Flowtime factor (1-β)	0,5

Table 7. Processing time matrix.

Resource	Type of product 1	2	3	4	5	6	7	8	9	10	11	12
1	1000	1000	1000	1000	50	47	39	43	10	12	15	13
2	1000	1000	1000	1000	33	39	32	25	22	25	23	28
3	15	12	16	13	1000	1000	1000	1000	49	46	44	47
4	16	15	11	18	1000	1000	1000	1000	31	38	35	37
5	22	23	27	30	13	11	10	16	1000	1000	1000	1000
6	48	44	49	50	19	17	16	20	1000	1000	1000	1000

6.10 Results

Using the corresponding simulation parameters, we obtained the results shown in figure 32. Each of the profiles corresponds to the interest variable measurement according

to the architecture of the application developed. So, figure 32-1 shows the results for the total reward. Figure 32-2 shows the results for the machine utilization. Figure 32-3 shows the results for the mean flow time. And figure 32-4 shows the results for reward function associated with the flowtime. Finally, the different results are analyzed in the following sub-section from technical and managerial perspectives.

6.11 Analysis of results and discussion

Through the SLR we found that using AI to solve the scheduling problem brings advantages related to structuring an appropriate, tailored, and systematic method capable of achieving a scheduling objective. We found different benefits in terms of less human intervention, effective ways, and quality of the solution for the analyzed manufacturing system when solving the scheduling problem. Besides, from a managerial point of view, we found that applying a strategy to solve the scheduling problem showed improvements in a multi-dimensional correlated way. This means that improving one dimension from the set of performance variables carried improvements in other dimensions (i.e., makespan reduction relates costs, lead time, resource usage, and profitability).

Bearing in mind what SLR reported and the main objective of this thesis, we analyzed the obtained results from a technical and managerial perspective. The technical view is aimed to discuss the linear combination for the reward function that best describes the solving process of the problem and the quantitative improvements obtained when solving the problem (scheduling) through the tool developed.

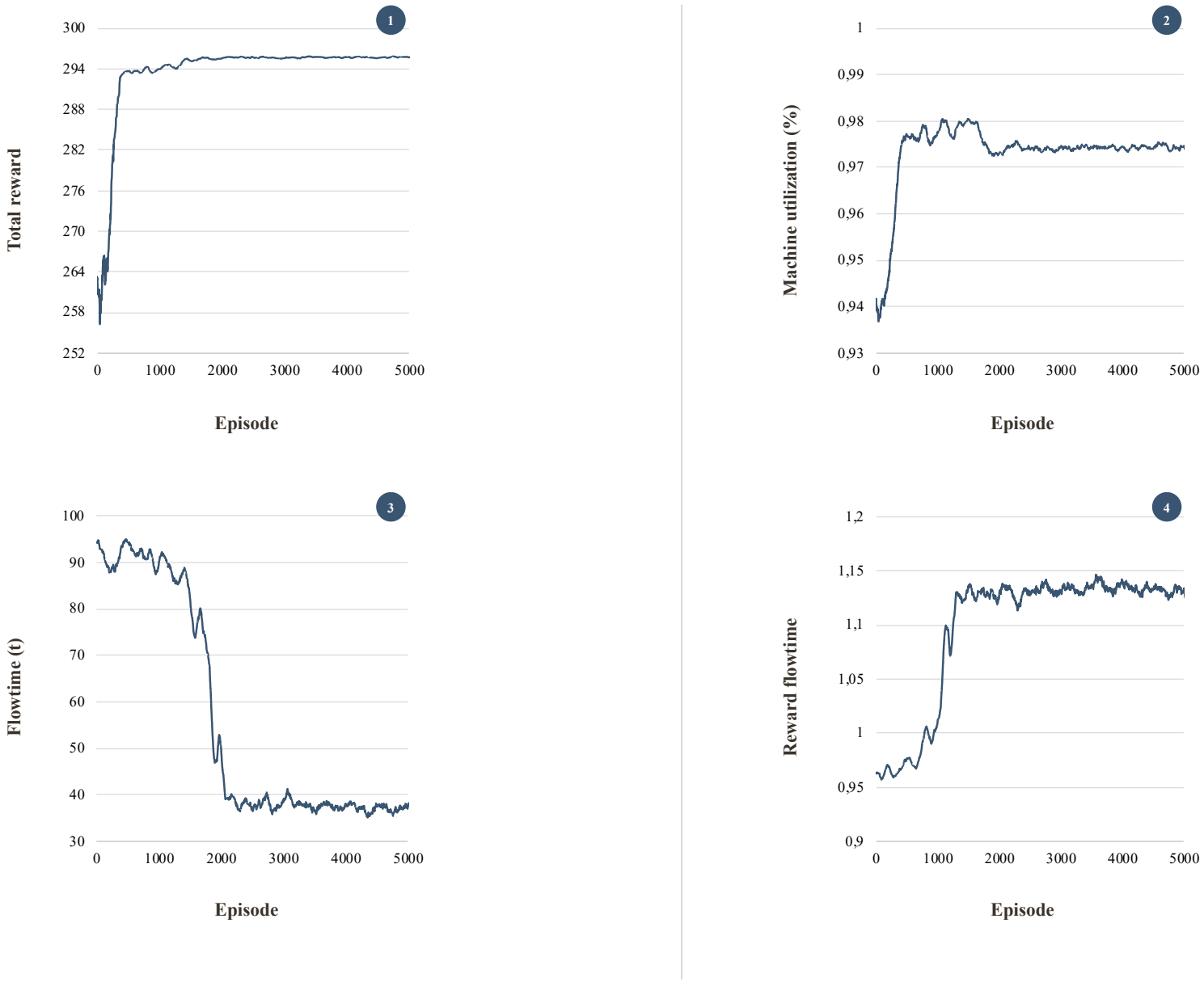

Figure 32. Simulation results. 1) Total reward. 2) Machine utilization. 3) Flowtime. 4) Reward flowtime.

So, we present the comparison between the results when integrating a RL technique to a well-known dispatching rule (SPT) and the results when this integration is not applied. Then, we present a managerial perspective analysis that considers a trade-off between the technical results, the advantages/disadvantages of integrating an AI technique to solve the scheduling problem, and the shortfalls of the tool's architecture proposed in this thesis.

6.11.1 Technical perspective analysis

As mentioned before, figure 32 shows the different results from the simulation run. First, we analyzed the coherence of the results regarding the reward function, which was set as the linear combination of the machine utilization and the flowtime. Then, we analyzed the performance of the results obtained with the tool developed through benchmarking with the solution given by the SPT dispatching rule.

Reward function

Considering the desired objective for the components that shape the reward function (maximization of the machine utilization and minimization of the flow time), we think the results that better explains the agent behavior when solving the scheduling problem must have the following criteria:

- The total reward should increase from episode to episode and be constant for late episodes. In this manner, we can affirm that the agent is learning to make better decisions through experience, and the solution converges.

- As the contribution developed is aimed to enhance the SPT dispatching rule, it is expected to obtain high values for machine utilization since this rule prevents clogging machines with long-duration jobs by giving preference to tasks with shorter processing times (Altendorfer, Kabelka & Stocher, 2007). Besides, this indicator is a counterweight for the flowtime, which is partially a contradicting objectives respect to the utilization, since the utilization is focusing on resources and the flowtime on orders (Kuhnle, Schäfer, Stricker & Lanza, 2019). So, it prevents biased decisions made by the agent due to the minimization of the flowtime. Therefore, it assures that the agent assigns jobs to the resources during the timeframe of the simulation.
- The flowtime obtained in each episode must decrease from episode to episode. This is an expected behavior related to the architecture of the reward function and the flowtime minimization.
- The reward associated with flowtime must increase from episode to episode. This behavior guarantees that the agent considers the relation between processing time and type of product when assigning jobs to resources, avoiding waiting times for the jobs, minimizing the makespan while processing jobs. Best the assignment that minimizes the waiting time, the higher the reward.

Figure 32-1 shows the total reward profiles obtained from the simulation. It was possible to observe that the total reward increased after an unstable initial learning phase.

Following episode 2000, it showed a steady-state behavior from episode to episode. This experiment presented the highest difference between the total reward for the transient and steady states. Figure 32-2 shows the machine utilization profile that resulted from the simulation. This indicator increased over time (expected behavior), and the obtained profile contained defined transient and steady states after the episode 2000. Figure 32-3 shows the resulting mean flowtime profile. The reward function used did not favor the minimization of the flowtime, but a decreasing profile was obtained. Therefore, it is possible to state that the agent is learning. Besides, it was possible to observe a defined transient and steady-state for the values obtained over time. Lastly, figure 32-4 shows the reward associated with the flowtime that resulted from the simulation. The obtained profile showed an increasing behavior, where it was possible to identify a steady state after 2000. In conclusion, this experiment's results are consistent in all the measured dimensions as explained in this section, and they fulfill the requirements stated in the main objective of this thesis.

Performance analysis

Considering that the results from the simulation were consistent and they explained in a good way the scheduling process made through a RL technique, we benchmarked the results. To do this, we considered machine utilization and flowtime obtained with the tool developed and the results obtained when SPT rule is applied directly over the same set of jobs to schedule.

The parameters used for solving the problem through the mentioned dispatching rule were:

- Total jobs: 500
- Jobs per episode (sample): 100
- Time units per episode: 300
- Total episodes: 5000
- Process time matrix shown in table 7.

Figure 33 shows the scheduling results obtained through SPT rule and the simulation run using the contribution developed. For making the comparison, we considered the results from our contribution of the last 2500 episodes, as shown in figure 33 (where convergence is achieved). In the case of SPT, we took the last 2500 points of the available data. Table 8 shows the comparison between results.

Table 8. Result comparison between experiment 1 and SPT rule.

Result	Experiment		Change
	Experiment 1	SPT	
Average machine utilization (%)	97,4%	91,8%	6,1%
Standard deviation	0,05%	2,2%	-97,8%
Coefficient of variation	0,05%	2,4%	-97,9%
Average flowtime (t)	37,9	82,2	-53,9%
Standard deviation	1,1	10,62	-89,6%
Coefficient of variation	2,9%	12,9%	-77.5%
Simulation time	89877,02	11346,4	692,1%

In production environments, one of the natural conflicts is to obtain a high throughput at a low level of WIP. At the same time, the level of WIP and the used dispatching rule determine the maximum utilization of the equipment in the system (Jones, 1973). Considering the Little's law (where the average work in process (WIP) equals throughput times flowtime), the information reported in Altendorfer, Kabelka & Stocher (2007) (high utilization of resources is obtained when flowtime is used as performance criteria for optimization), and the results obtained in the table 8, we could highlight that:

- There was a slight improvement in the average machine utilization (6.1%). When the scheduling problem is solved through SPT rule, the machine utilization resulted in 91.8%. When the problem is solved using the solution proposed in this thesis, it resulted in 97.4%. However, the standard deviation decreased 97.8%, which reduced the coefficient of variation by 97.9%.
- The average flowtime presented a reduction of 53.9%, with a decrease in the coefficient of variation of 77.5%. Besides, the flowtime reduction was expected since the contribution developed was designed with a reward function that minimizes this performance indicator.
- Compared to the solution provided by SPT rule, the contribution developed in this thesis presented an increment in the simulation time of 692.1%. This is because RL techniques require a large number of samples for the learning process.

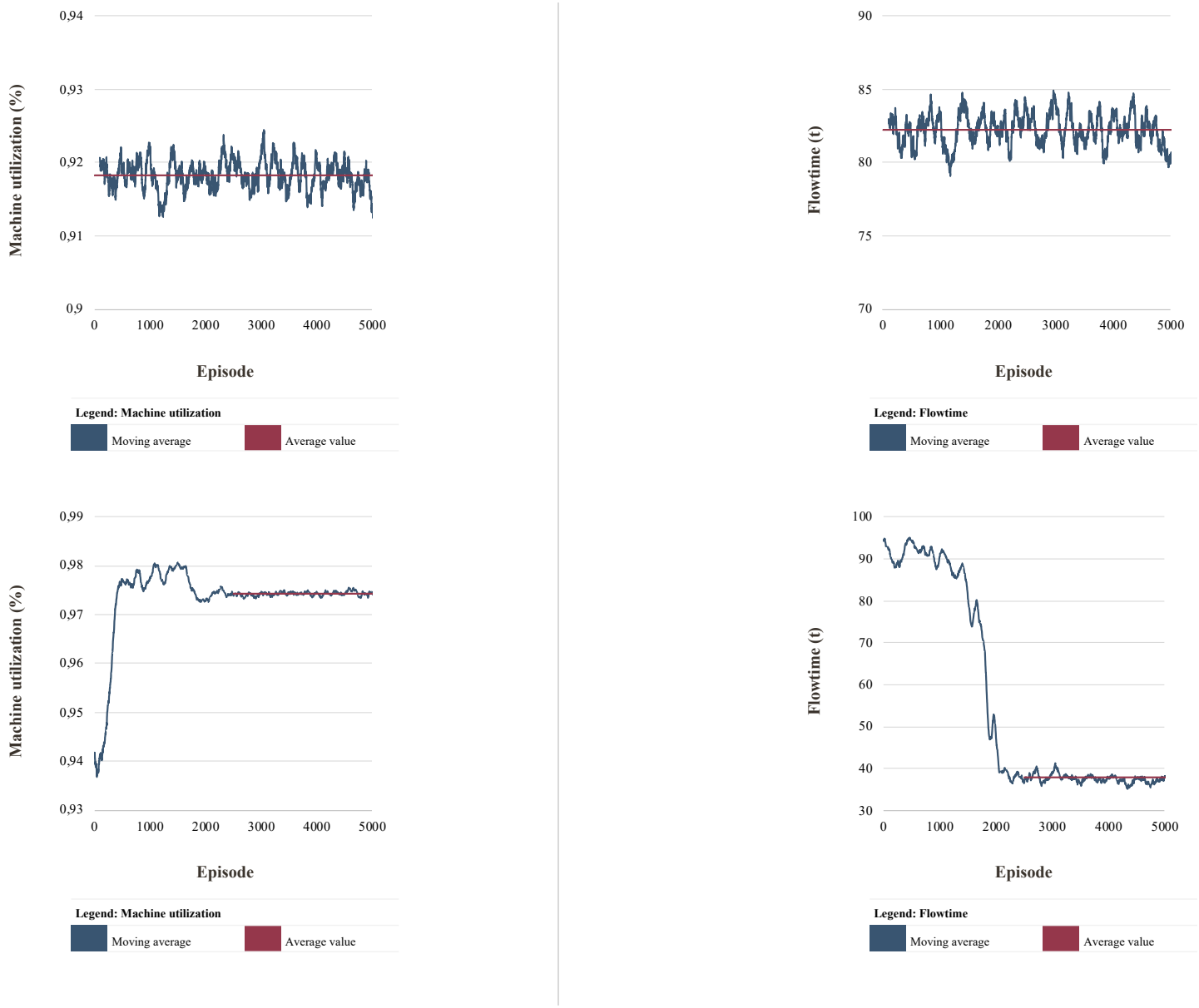

Figure 33. Results benchmarking. Top: SPT. Bottom: RL method.

6.11.2 Managerial perspective analysis

Due to the resulting changes in performances between the solutions provided by the contribution developed in this thesis and SPT rule, an analysis from a managerial perspective is necessary. To do so, it is required to think about the results holistically and understand the trade-off between them. Figures 34 shows graphically the result obtained from the comparison made in the technical analysis.

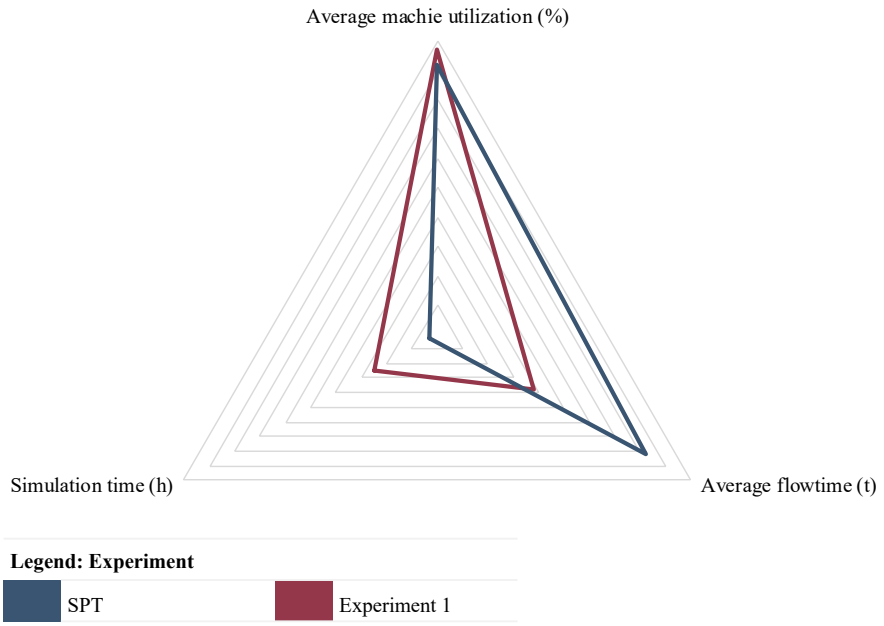

Figure 34. SPT rule vs contribution developed performance comparison.

Since the problem boarded in this thesis relates to manufacturing systems and the obtained results showed improvement in some dimensions while worsening others, there

is an opportunity to analyze them from a manufacturing operations management perspective and their relationship with the business strategy. This analysis aims to understand better the impact of the results obtained on the business objectives and not only see them as technical parameters. Typically, these objectives are enclosed in five performance dimensions related to: i) time, ii) price (cost), iii) quality, iv) flexibility, and v) service. Moreover, it is useful to state that, according to Jönsson (2012), success in all these dimensions is not only impossible but sometimes useless. Following, we present the strategy-operation analysis based on the obtained results using as an example the aerospace industry.

Average flowtime

As stated by Deloitte (2016), in industries that belong the aerospace market (large capital expenditure and intensive labor), the company strategy is thought in a way to satisfy different market needs:

- **Bespoken development:** products requested by the customer with specific requirements (large investments, high quality, small numbers of units).
- **Highly tailored product line:** products that benefit from cost and speed-to-market of an existing product line, but they required modification to satisfy a specific need.

- **At-rate product line:** products manufactured in a single way and delivered with little to no tailoring (low cost, delivery schedule stability, high production volume).
- **Global-scale and scope:** products where price is the primary decision factor. These products do not required tailoring, defined designs, low cost, and very high volume.

So, linking the results for flowtime that resulted from scheduling through SPT rule and the contribution developed in this thesis, with the strategies followed by aerospace industries, it is possible to state that:

- For those organizations that based their strategy on bespoken development or highly tailored product line, the minimization of the flowtime impacts positively performances related to time (delivery speed) and cost (inventory carrying cost, revenue opportunity cost, and variable tooling cost) (Chao & Graves, 1998). In this way, the business gains competitive advantage.
- For those organizations that based their strategy on at-rate product line or global-scale and scope, the minimization of the flowtime impacts positively (in a higher level) the number of produced units, improving performances related to cost (benefits from economies of scale) and

service (higher availability of goods) (Slack & Brandon-Jones, 2018). So, the business could gain competitive advantage.

Figure 35 shows the changes in the performance dimensions related to each of the strategies described above and that industries in the aerospace market follow.

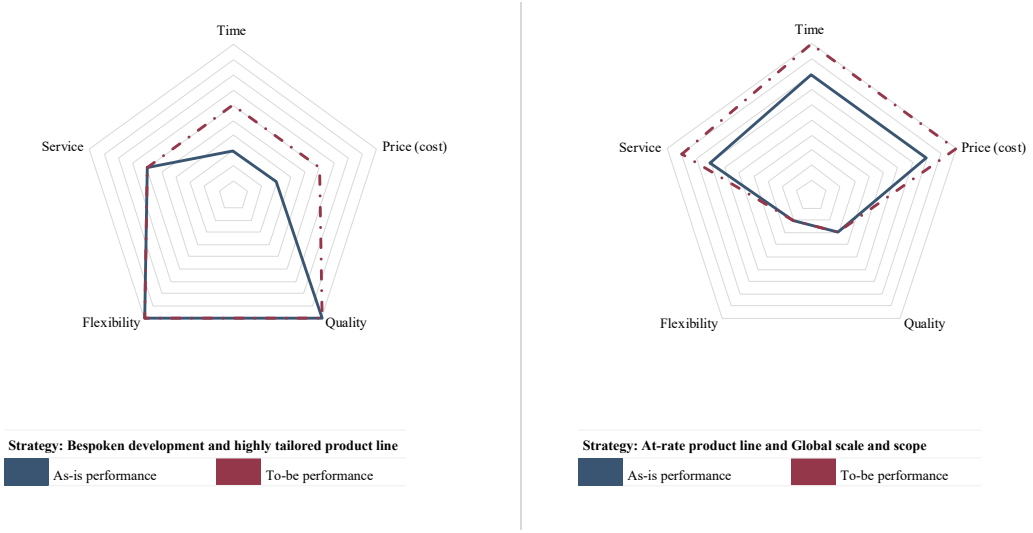

Figure 35. Performance dimension analysis impact (flowtime and jobs done).

Machine utilization

Considering the conflict high throughput - low level of WIP, the throughput maximization is essential in industries with high fixed costs for machines and equipment since it permits achieving a satisfactory level of return on shareholders' equity (Cigolini, 1999). Arnaboldi, Azzone & Giorgino (2014) stated that the fixed cost is allocated to each

of the manufactured units. So, as the number of units produced increases, the fixed cost per unit decreases, and the profit increases. From the comparison of the results obtained using SPT rule and the contribution developed in this thesis, we obtained an increment of 6.1% in machine utilization. This implies a higher number of units produced, which would decrease the fixed costs, improving the performance dimension related to price (cost). In this way, through the solution proposed, we can state that cost reduction is possible and an increase in profits is achievable.

As done before, if we tie this analysis to the strategy-operation perspective and we take as example the analysis made for the aerospace industry, it is possible to affirm that:

- For those organizations that based their strategy on bespoken development or highly tailored product line, the increment of the machine utilization impacts positively performances related to cost (reduction of fixed costs). However, since these strategies do not rely on quantity produced, increasing the utilization could not represent a gain of competitive advantage.
- For those organizations that based their strategy on at-rate product line or global-scale and scope, the increment in the machine utilization has a higher positive impact in performances related to cost (reduction of fixed costs), and service (availability of units produced) since these strategies are quantity dependent. So, the business has an enormous potential to gain competitive advantage.

Figure 36 shows the changes in the performance dimensions related to each of the strategies described above and that industries in the aerospace market follow.

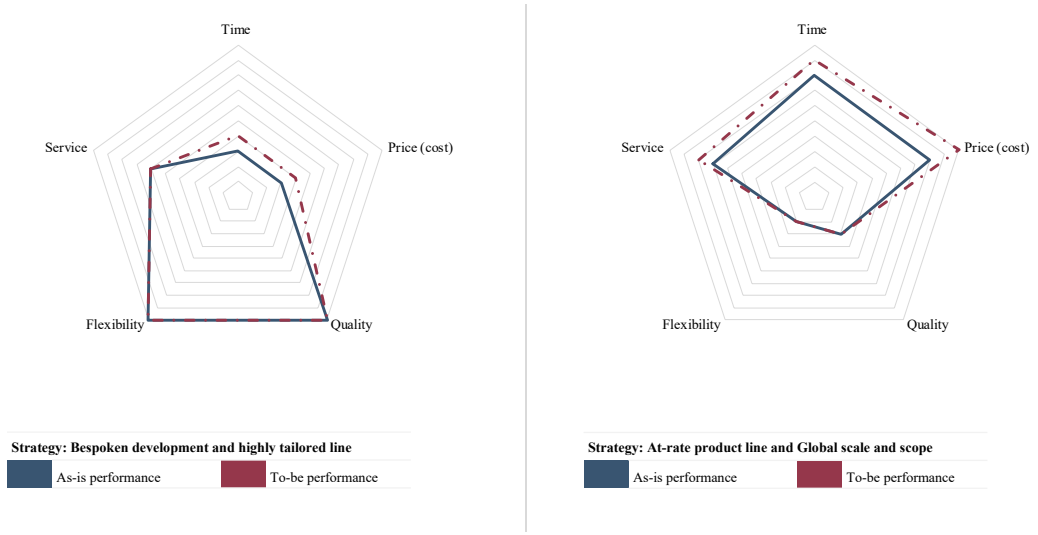

Figure 36. Performance dimension analysis impact (machine utilization).

Simulation time

The integration between a function as PPC and AI techniques implicates performance improvements, but at the same time an increment in computational costs. According to Sutton & Barto (2018), there is a computational mega-trend where effective computation per dollar increases exponentially, with a doubling time of 18-24 months. Besides, this trend has held for the last sixty years and will continue for the foreseeable future. However, Sutton & Barto (2018) also stated that computation becomes cheaper over time. Still, it is not free, and this cost has relevant importance since the demand is essentially infinite, and the performance will depend on how efficiently computation is used. Gómez-Carmona, Casado-Mansilla, Kraemer, López-de-Ipiña, & García-Zubia

(2020) explored the computational cost of ML classification techniques, they analyzed all the factors affecting the classification process, and they made an empirical evaluation of their impact in terms of performance and cost. They based their analysis on the limitations in the computational capabilities when executing ML techniques and the accuracy needed for solving problems through this AI technique, a characterization known as the accuracy–cost trade-off. For the case they explored, the factors that affected this trade-off were: i) the platform on which the classification is performed and the set-up, ii) the changing rate of the database to analyze, iii) the size of the database to analyze, iv) the number of features to extract from the data, v) the machine learning algorithm used. Figure 37 depicts the representation of the factors described above for the cost–accuracy trade-off.

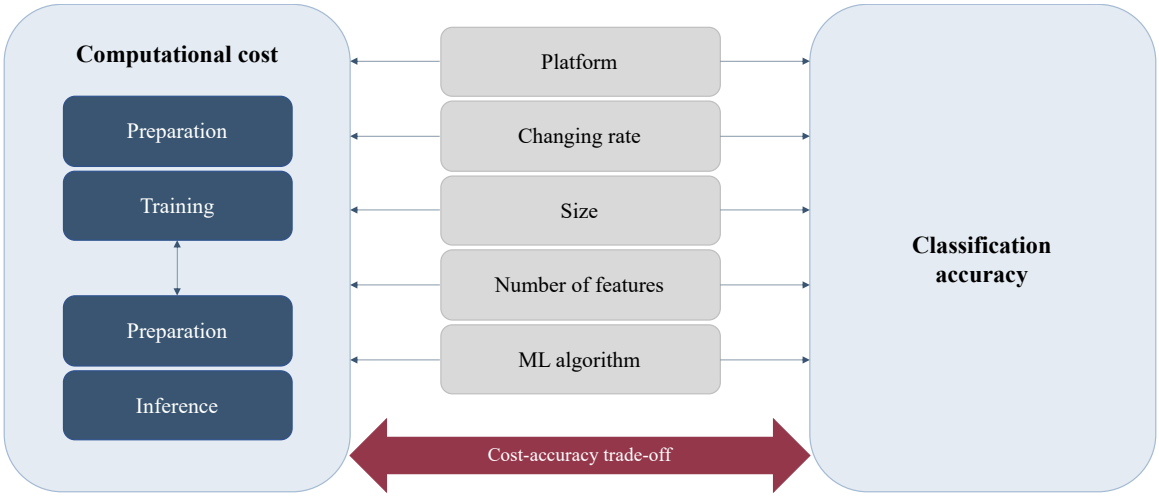

Figure 37. Factors affecting the cost-accuracy trade-off.

Since the contribution proposed in this thesis is based on a RL technique, the computational cost increments naturally. However, computer cost optimization based on the fundamentals of the algorithm is possible, but it is out of the scope from this thesis.

Chapter 7 – Conclusions, limitations, and future work

In this thesis, a RL model was applied to solve the scheduling problem for a specific manufacturing system. To understand the advantages that AI approaches bring when integrating with scheduling function in manufacturing environments, we first structured the research based on a SLR. Through this method, we found that the mentioned integration has been done in several manufacturing systems with different features (dynamic/static, discrete/continuous, etc.), considering various implementation features (industry types, number of machines in the system, AI approach, etc.), and analyzing possible performance improvements (cost, flexibility, productivity, etc.). Besides, this systematic methodology permitted us to conclude that this integration generates different benefits in terms of less human intervention, faster methods, and a goodness of the solution for the analyzed manufacturing system when solving the scheduling problem. Moreover, it permitted to state that the integration between AI and scheduling always implies performance improvements in cost, productivity, energy usage, lead time, and many others. On top of these conclusions, the SLR opened the opportunity for exploring a new way to solve the problem addressed in this thesis since this methodology allowed to figure out gaps and trends from available literature where the scheduling problem is solved through an AI approach. Given the found gaps and trends, we adopted ML as an AI approach for solving the scheduling problem addressed, considering as objective the minimization of the flowtime and maximization of the machine utilization.

Once the AI method was established and the research objectives defined, we made a comprehensive revision of the literature where ML techniques were used to solve the scheduling problem in a manufacturing environment. From this revision, we found that RL was used when the set problem involved flexibility to face changes in the system and reduced the computational cost, requiring conditions to apply the developed solution into a real-world problem. In this way, we constructed an application on Python based on a RL method that solves the scheduling problem for a parallel machine system that processes jobs with a defined processing time, with intermittent job order arrivals. The way to better understand the results and improvements obtained was by comparing the resulting machine utilization and flowtime between the solution that provides SPT dispatching rule and the tool developed in this thesis.

From a technical perspective, compared to the SPT dispatching rule, we obtained a slight improvement in the average machine utilization (6.1%). However, the standard deviation decreased 97.8%, which reduced the coefficient of variation by 97.9%. Besides, the average flowtime presented a reduction of 53.9%, with a decrease in the coefficient of variation of 77.5%. Lastly, the contribution developed in this thesis presented an increment in the simulation time of 692.1% due to the required large number of samples for the learning process in the RL techniques.

On the other hand, from a managerial perspective, we concluded that the obtained technical results must be overlapped with the organization's business objectives that own the manufacturing system to determine if they are beneficial or harmful. For the obtained

result in flowtime reduction, if an organization based its business objectives on satisfying market needs related to unique products with high quality, the minimization of the flowtime impacts positively performances related to time (delivery speed) and cost (inventory carrying cost, revenue opportunity cost, and variable tooling cost). Moreover, if an organization based its business objectives on satisfying market needs related to massive production where the cost is a competitive advantage, the minimization of the flowtime has a higher positive impact over the number of units produced, improving performances related to cost (benefits from economies of scale) and service (higher availability of goods). For the resulted increment in machine utilization, we considered the changes that impact costs. So, we concluded that for an organization that based its business objectives on satisfying market needs related to unique products with high quality, the increment of the machine utilization impacts positively performances related to fixed cost. However, since this strategy does not rely on quantity produced, the increment in the utilization could not represent a competitive advantage. Contrary, for organizations that based their business objectives on satisfying market needs related to massive production, machine utilization positively impacts the cost (reducing the fixed cost) and service (availability of units produced). In conclusion, it was possible to state that there are benefits that arise from integrating an AI-based method with a traditional company function. However, they must be analyzed technically (where changes in performances are shown) and from a managerial perspective (where the impact on business objectives is observed).

As limitations, we considered two points that could not permit using the proposed solution in this thesis in a real-world case. The first point is that we only considered one among many events in a real-world manufacturing scenario (intermittent job order arrivals). So, the algorithm was trained to face only this type of change successfully. The second point is related to the increment of the simulation time this type of algorithm requires. Although the contribution developed faces rapid changes related to the arrival of jobs, the training needs a lot of time. If we considered more changing factors (maintenance, quality issues, cancelation of order, etc.), the simulation time (including the training) would be higher than the result obtained in this thesis. Therefore, for future work, applying AI approaches to solve scheduling problems must be explored more and more to improve technical performance and fulfill managerial terms (i.e., solving the scheduling problem for job shops).