# Optimizing Job-Shop Scheduling in Manufacturing Environments Using AI: A Hybrid Approach with Genetic Algorithms and Reinforcement Learning


Tharindu DeSilva
Responsible for model development and wrote sections 1 and 3

Gurudeep Haleangadi Nagesh
Responsible for software development and wrote sections 2 and 3

Ranjitha Umesh
Responsible for maintaining documentation and wrote sections 1 and 4

Dyuthi Nagaraja Kedilaya
Responsible for analysis of results and contributed to sections 2 and 4



## ABSTRACT

The paper focuses on optimizing job-shop scheduling within the dynamic and complex landscape of Industry 4.0 manufacturing environments. It introduces a novel approach that harnesses the synergy of Genetic Algorithms (GAs) and Reinforcement Learning (RL), particularly Proximal Policy Optimization (PPO), to enhance scheduling efficiency. Our research addresses critical challenges such as efficient task allocation, minimization of makespan, and effective management of tool changeover times. The GA component of our model is responsible for optimizing task sequences in each job, taking into account the constraints of tool availability and lifespan. Concurrently, the RL model strategically distributes jobs across various machines, aiming to reduce the total completion time. The effectiveness of this integrated approach is demonstrated through simulations that closely replicate real-world manufacturing settings. The results reveal a marked improvement in both scheduling efficiency and resource utilization. This paper contributes to the evolving field of intelligent manufacturing systems by offering an innovative, adaptable job-shop scheduling solution, aligning with the objectives of Industry 4.0 to achieve greater automation and process optimization.



## KEYWORDS

Job-Shop Scheduling, Genetic Algorithms, Reinforcement Learning, Proximal Policy Optimization, Artificial Intelligence in Manufacturing, Makespan Minimization


## 1 INTRODUCTION

In today's complex manufacturing setups where robots, AGVs, and machinery collaborate, efficient scheduling is essential, especially when tool changes are required due to varying production demands or tool wear. Our research introduces an innovative approach that combines Genetic Algorithms with Reinforcement Learning, specifically Proximal Policy Optimization, to address these challenges.

Our method focuses on optimizing task sequencing and resource allocation to streamline production workflows. By considering critical factors such as tool lifespan and job-specific requirements, we minimize downtime and make the most of available resources.

What sets our approach apart is its comprehensive strategy towards job-shop scheduling, encapsulated by three key strengths:

(1) **Consideration of Attributes in Complex Job-Flow Environments:** We account for various attributes crucial to efficient scheduling, enhancing overall production efficiency.
(2) **Innovative Combination of Techniques:** The unique integration of Genetic Algorithms and Reinforcement Learning offers a powerful and innovative approach to tackling scheduling challenges.
(3) **Strategic Management of Tool Changeovers:** We minimize downtime associated with tool changes and optimize tool usage, significantly boosting productivity in manufacturing setups.

An illustration of a real-world job-shop environment is presented in Figure 1.

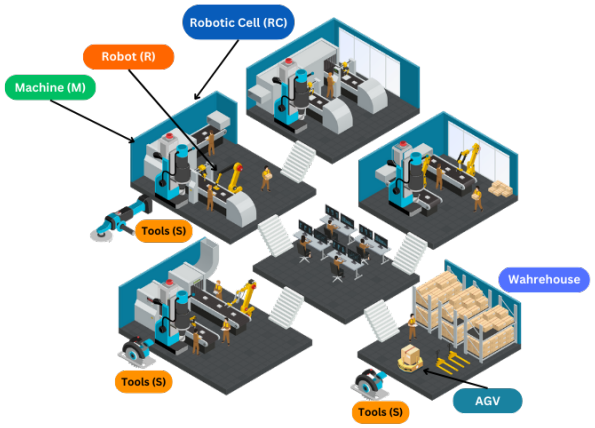



**Figure 1: Realworld Jobshop Environment**

## 2 LITERATURE REVIEW

Job-shop scheduling is a complex domain with various potential approaches, each addressing specific aspects of the manufacturing process. Traditional methods, while effective in static environments, struggle in the dynamic and uncertain landscapes of modern industrial settings [1]. The complexity of industrial job-shop scheduling stems from numerous factors, including variability in task durations, machine availability, and the need for real-time adaptability.

Papers [2] and [3], delve into these complexities, emphasizing the challenges in achieving high autonomy in production cells with considerations like material flow and tool lifetime. Our project addresses these complexities by integrating AI techniques, specifically Reinforcement Learning (RL) and Genetic Algorithms (GA). RL has been increasingly used in job-shop scheduling for its ability to make real-time decisions and adapt to changing environments [4, 5]. GAs, on the other hand, have been effectively employed to optimize scheduling solutions, focusing on resource utilization and minimization of makespan [6]. The possibility of combining RL and GA presents a novel approach to job-shop scheduling. While RL provides dynamic decision-making capabilities, GAs offer robust optimization strategies [7, 8]. This combination has the potential to address the unique challenges of industrial job-shop scheduling, as highlighted in [2] and [3]. Our approach aims to fill the gaps in current job-shop scheduling practices by offering a system that is not only efficient but also adaptable to the intricacies of modern manufacturing processes. By employing both RL and GA, our method enhances scheduling efficiency, reduces production time, and achieves a higher level of precision in manufacturing operations.

## 3 METHODOLOGY

This research introduces a novel methodology that synergizes Reinforcement Learning (RL) and Genetic Algorithms (GA) to optimize job-shop scheduling within a high-precision manufacturing context. Our approach aims to minimize the overall completion time (make-span) of all jobs while reducing tool changeover times and extending tool lifetimes.

In the initial phase, we employ an RL-based agent to allocate jobs among available machines, focusing on optimizing the preliminary schedule based on processing times and machine availability. This step ensures that jobs are assigned in a manner that preliminarily minimizes the make-span without yet considering tool-specific constraints.

Subsequently, we enrich the job scheduling model with meta-information, such as task, associated tools, and other relevant parameters (task duration, robotic cell). This enriched model serves as the input for the GA-based optimization layer, which refines the initial schedule. The GA layer specifically targets the minimization of tool changeover times and the efficient management of tool lifetimes, all while maintaining or improving the overall makespan.

Through this integrated approach, the system dynamically adapts to the complexities of job-shop scheduling in high-precision manufacturing environments, leading to more efficient and effective production processes. The overall methodology is depicted in Figure 2.

### 3.1 Reinforcement Learning for Initial Job Distribution

**Objective**: To distribute jobs across available machines in a way that minimizes the makespan, disregarding tool changeover times in the initial phase.

**Model and Implementation**:

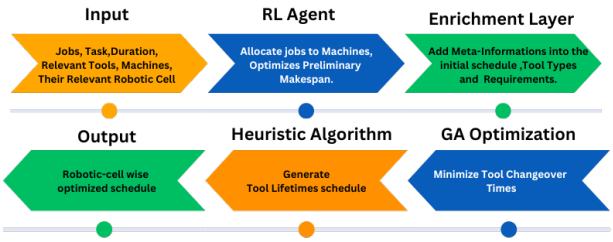



**Figure 2: Overall Methodology**

- **Reinforcement Learning Model**: Utilizing the Proximal Policy Optimization (PPO) algorithm for the initial job distribution.
- **State (S)**: Represents the current distribution of jobs across machines. For example, $S_i = \{(J1, M2), (J2, M1), \ldots\}$ indicates the allocation of jobs to machines.
- **Action (A)**: Involves reassigning a job to a different machine, such as moving Job 3 from Machine 1 to Machine 2.
- **Reward (R)**: Calculated as $R(S, a)$ = Previous Makespan – New Makespan, rewarding actions that reduce the overall completion time.
  For improve the flexibility , Given a target makespan $M_{target}$ and a tolerance level $T$, the reward function $R$ for a reinforcement learning scheduler can be defined as:

$$R = \begin{cases} \alpha & \text{if } |M_{current} - M_{target}| \leq T \\ -|M_{current} - M_{target}| & \text{otherwise} \end{cases}$$

  Where:
  - $R$ is the reward given to the agent.
  - $M_{current}$ is the makespan of the schedule proposed by the agent.
  - $M_{target}$ is the target or desired makespan, which the agent aims to achieve or undercut.
  - $T$ is the tolerance level for the difference between the current and target makespans.
  - $\alpha$ is a positive constant, rewarding the agent for achieving a makespan within the tolerance level of the target.

  This reward structure incentivizes the agent to generate schedules with makespans close to or better than the target, within a defined tolerance, thereby improving scheduling efficiency.

**PPO Algorithm Overview**: The algorithm iteratively optimizes the policy $\pi(a|S)$, representing the probability of assigning a job to a machine based on the current state. The model is trained using real-world data, adapting to dynamic manufacturing conditions and efficiently assigning jobs.

### 3.2 Genetic Algorithm for Task Sequence Optimization

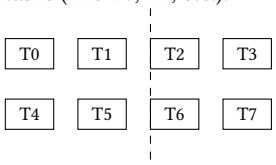
**Objective**: To fine-tune the task sequences within each job, focusing on reducing tool changeover times and optimizing the final makespan.

| Author(s) | Algorithm Used | Main Target | Robotic Cell | Robot Collaboration | Tool Breakdown (Planned) |
|---|---|---|---|---|---|
| Author 1 et al. [? ] | ERL | Sequential Decision-making | × | × | × |
| Author 2 et al. [? ] | GA for RL | Drone Delivery Optimization | × | ✓ | × |
| Author 3 et al. [? ] | Species-based GA | RL Problem Efficiency | × | × | × |
| Author 4 et al. [? ] | GA for DNNs | Training DNNs for RL | × | × | × |
| Author 5 et al. [? ] | ES | Scalability in RL | × | × | × |
| Author 6 et al. [? ] | GA + DRL | Flow Shop Scheduling | × | × | ✓ |
| Author 7 et al. [? ] | GA for DNNs | Deep RL Problems | × | × | × |

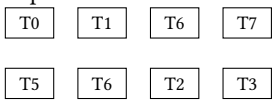
**Table 1: Summary of Studies on GA and RL**

**Implementation Details**:

- **Chromosomes**: Represent potential manufacturing schedules, where each chromosome lists tasks assigned to different machines.
- **Crossover (SwapTasks)**: A process that randomly selects two machines and exchanges tasks between them to generate new schedules.
  **Parent Chromosomes:** In genetic algorithms, parent chromosomes represent current solutions, each composed of tasks (like T0, T1, etc.).




  **Offspring Chromosomes After Crossover:** Offspring chromosomes are new solutions formed by mixing parts of two parents, aiming to combine their best features for improved solutions.




- **Selection**: Based on a fitness score that considers task dependencies and machine assignments, selecting schedules that offer a balance between short makespan and efficient tool utilization.

**Fitness Function**: The fitness of a chromosome is calculated to balance the makespan and tool changeover times, defined as follows:

$$\text{Fitness}(C) = \frac{1}{\text{Makespan}(C) + \lambda \times \text{ToolChangeoverTimes}(C)} \quad (1)$$

where $C$ represents a chromosome, and $\lambda$ is a weighting factor that balances the importance of makespan and tool changeover times.

## 3.3 Integration and Further Optimization

**Objective**: To leverage heuristic methods for enhancing overall manufacturing operations, considering tool lifetime and AGV optimization.

**Tool Lifetime and AGV Optimization**:

- **Tool Lifetime Finishing Time Calculation**: Calculating the expected end-of-life for each tool based on its usage in the optimized schedule.
- **AGV Optimization**: Optimizing routes and schedules of AGVs based on tool lifetime data, aiming to minimize downtime and improve operational efficiency.

## 3.4 Algorithms

**Data:** Jobs ($J$), Job Durations ($D$), Machines ($M$)
**Result:** Optimized Schedule ($S_{opt}$)
Initialize Reinforcement Learning Environment ($Env_{RL}$) with $J$, $D$, $M$;
Initialize Genetic Algorithm Population ($Pop_{GA}$);
`/* Iteratively train the model and refine job durations */`
**for** *epoch = 1 to Total Epochs (E)* **do**
  Generate random job durations ($D_{rand}$);
  Determine target makespan ($T_{ms}$) using GA with $D_{rand}$, $M$;
  Configure $Env_{RL}$ with new $D_{rand}$ and $T_{ms}$;
  Train Reinforcement Learning Model ($Model_{RL}$) within $Env_{RL}$;
**end**
Extract optimized schedule ($S_{opt}$) from $Model_{RL}$;

**Algorithm 1:** Enhanced Job Scheduling with RL and GA

Algorithm 1 combines reinforcement learning (RL) with a genetic algorithm (GA) to optimize job scheduling. The RL environment is

initialized with jobs, their durations, and the machines available. The GA is used to determine an ideal target makespan based on random job durations, guiding the RL model's training towards efficient scheduling.

**Data:** Jobs ($J$), Job Durations ($D$), Machines ($M$)
**Result:** Balanced Schedule ($S_{bal}$)
Initialize Population ($Pop$) with random job-to-machine assignments;
/* Evolve population over generations for optimal scheduling */
**for** *generation = 1 to Generations (G)* **do**
    Calculate Fitness ($F$) for each individual in $Pop$ considering makespan and workload balance;
    Select individuals for reproduction ($Ind_{repro}$) via Tournament Selection;
    Generate Offspring ($Offspring$) through Crossover and Mutation of $Ind_{repro}$;
    Update $Pop$ with $Offspring$;
**end**
Identify the best individual ($Best$) in $Pop$ based on $F$;
Decode $Best$ into a readable schedule ($S_{bal}$);

**Algorithm 2:** GA for Balanced and Efficient Machine Scheduling

Conversely, Algorithm 2 leverages a genetic algorithm to allocate jobs across machines in a manner that ensures balance (i.e., no machine is overburdened) and efficiency (i.e., jobs are completed in the shortest possible time). It starts with a population of random job assignments and iteratively improves this population through selection, crossover, and mutation, guided by a fitness function that accounts for both makespan and workload balance. The best solution is then decoded into a clear and balanced schedule.

# 4 CONCLUSIONS

This study introduces an artificial intelligence (AI) scheduling system specifically created for job shop settings. The main goal of this system is to arrange tasks efficiently, taking into account the availability of tools, how tasks depend on each other, and the best use of resources. The research employs a genetic algorithm to organize the scheduling tasks and uses a reinforcement learning model to initially assign the jobs to machines. This combined approach results in a scheduling solution with the best possible duration for completing all tasks (optimal makespan), while also addressing various constraints within the genetic algorithm.

## 4.1 Key Findings

This section of the paper highlights the main findings from our study, which takes into account various real-world constraints in a manufacturing setting.

To address these constraints and optimize the manufacturing process, we incorporated a mix of reinforcement learning, genetic algorithms, and heuristic algorithms. The heuristic algorithms were particularly useful in managing the allocation of tasks between robots and machines, as well as in creating tool breakdown charts, which are essential for planning the maintenance and availability of tools[3].

## 4.2 Operational Constraints

During the study, several operational constraints in a real-world manufacturing scenario were considered [3]:

(1) Every machine (labelled as $M_i$, where $i = 1, 2, \ldots, M$) is capable of producing all types of parts (denoted as $J_n$, with $n = 1, 2, \ldots, N$).
(2) Any production order can be allocated to any of the machines.
(3) Each machine is limited to producing one part at any given time.
(4) Production periods for each machine must not overlap.
(5) All parts within a single order are identical and are processed one after another without interruption.
(6) Specific sets of tools ($T_i$, where $i = 1, 2, \ldots, T$) are available for use by all machines, but only one machine can use a set at any given time.
(7) The movement of raw materials and the delivery of finished parts are managed by Automated Guided Vehicles (AGVs).

.

(1) **Task and Machine Scheduling:** Our first scheduling shown in Figure 3, model prioritizes the efficient distribution of tasks across various machines, with a keen focus on minimizing tool changeover times and the overall makespan. This approach adeptly balances the need for operational efficiency with the complex requirements inherent to different machinery. The study employed reinforcement learning to initially assign jobs to machines. The genetic algorithm was then used to schedule the tasks of jobs across the machines. The figures below illustrate the task and machine schedules before and after optimization for 10 machines handling 10 jobs, each with a varying number of tasks; while some jobs consist of 4 tasks, others may have only 2 tasks. The scheduler takes into account the previously mentioned constraints to generate the schedule.
(2) **Robotic Cell Scheduling:** Our study's second scheduling model, depicted in Figure 5, focuses on robotic cells. Originally, the robots were only assigned the task of loading and unloading materials, which resulted in periods of inactivity. Reference [3] proposes an improved schedule that involves assigning additional tasks to the robot. To decrease the frequency of tool changes and enhance efficiency within the robotic cell, we allocate tasks that require the most frequently used tool to the robotic cell, especially if that tool is not shared among multiple cells. This strategy significantly cuts down on the time spent changing tools and ensures the robot operates at a higher level of optimization.
(3) **Tool Life Scheduling:** By using a heuristic algorithm, our third scheduling strategy proactively incorporates the life expectancy of tools to optimize maintenance and replacement timelines. This method not only anticipates when and on which machine a tool will require changing, but also coordinates with the Automated Guided Vehicle (AGV)

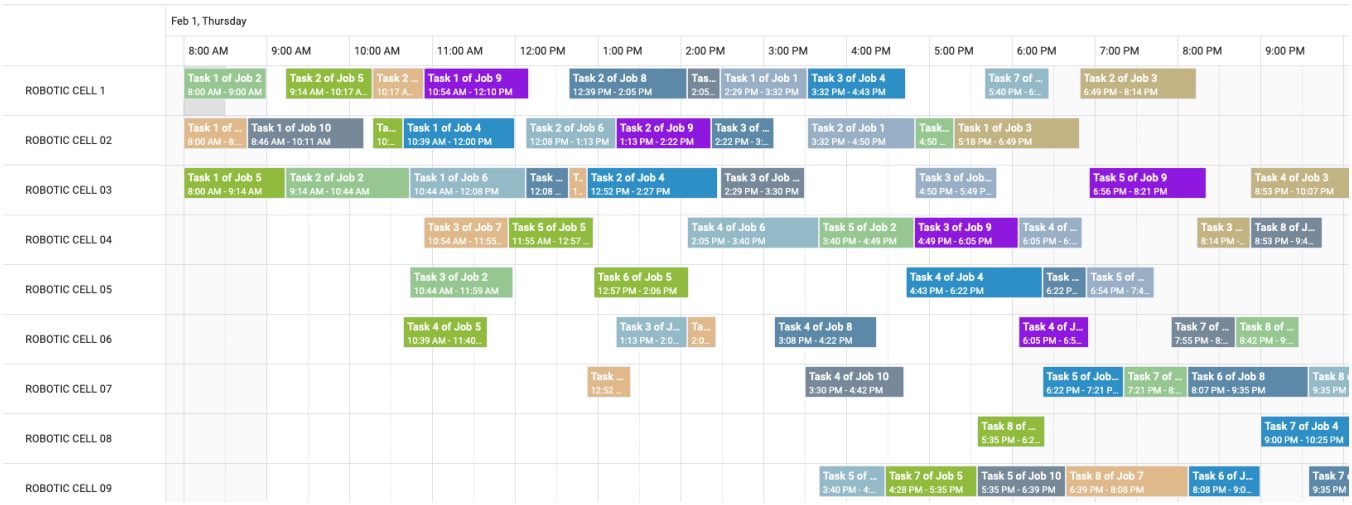



**Figure 3: Task and Machine before Scheduling**

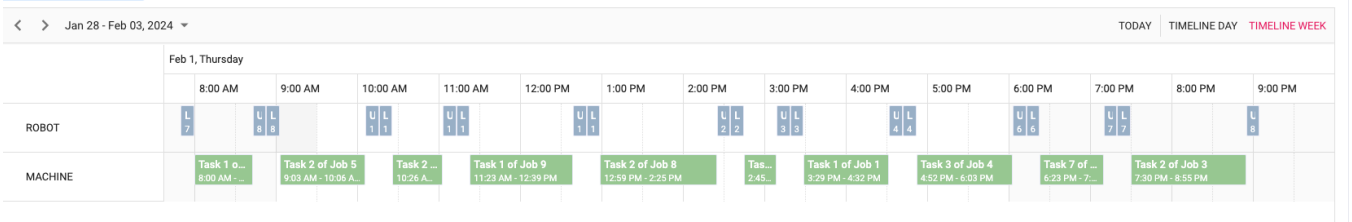



**Figure 4: Robotic Cell Scheduling Before.**

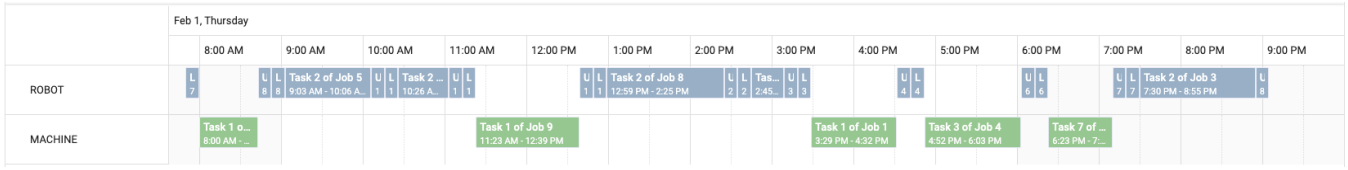



**Figure 5: Robotic Cell Scheduling after.**

system to ensure tools are delivered precisely when needed. This minimizes the downtime associated with tool changes, especially during unexpected breakdowns. The planned schedule, illustrated in Figure 6, is a significant step towards enhancing operational continuity and efficiency by seamlessly integrating tool life cycle management into the production process.

## 4.3 Implications and Contributions

This paper focuses on the optimization of job-shop scheduling within the dynamic and complex landscape of Industry 4.0 manufacturing environments. We introduce a novel approach that combines the synergistic potentials of Genetic Algorithms (GAs) and Reinforcement Learning (RL), particularly Proximal Policy Optimization (PPO), to advance scheduling efficiency. Our research tackles critical challenges in manufacturing, such as the efficient allocation of tasks, minimization of the overall completion time (makespan), and effective management of tool changeover times. The GA is tasked with optimizing task sequences for each job, mindful of the constraints related to tool availability and lifespan. Simultaneously, the RL model employs strategic job distribution across machines to minimize total completion time. The effectiveness of this integrated model is validated through simulations that accurately mirror real-world manufacturing scenarios, revealing significant improvements in scheduling efficiency and resource utilization. This paper enriches the field of intelligent manufacturing systems by proposing an innovative and flexible job-shop scheduling solution, which is in alignment with the objectives of Industry 4.0 for enhanced automation and process optimization.

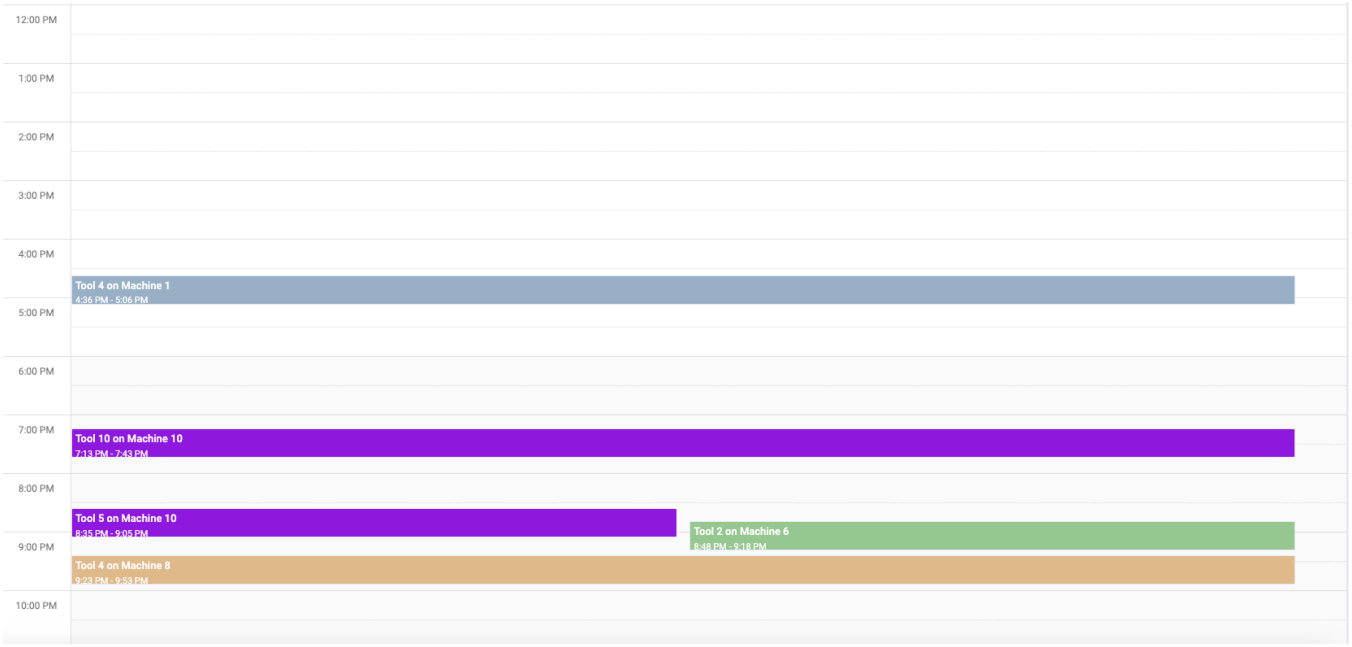



**Figure 6: Tool Life Scheduling.**

## 4.4 Future Work

The potential for further research in the area of dynamic job-shop scheduling is substantial, offering numerous promising avenues:

- **Adaptive Scheduling:** Developing reinforcement learning algorithms that adapt in real-time to unexpected events like unplanned breakdowns, workforce variability, and the introduction of high-priority jobs.
- **Resilience in Scheduling:** Examining methods for resilient scheduling that can quickly reconfigure and reschedule tasks in the face of new constraints and deadlines, ensuring continuous operation.
- **AI Innovations:** Investigating cutting-edge AI techniques to further enhance the flexibility and efficiency of job-shop scheduling, pushing beyond the current capabilities and addressing the evolving complexities of Industry 4.0 environments.

This research lays the foundation for sophisticated AI applications in manufacturing, steering the progression toward completely optimized systems that fully embrace the ideals of Industry 4.0.

## ACKNOWLEDGMENTS


- We are deeply grateful to Prof. Dr. Jan Schmitt and Mr. Eddi Miller for their expert guidance and supervision throughout the course of our project. Their knowledge and experience have been crucial for our research's success.
- Our thanks also extend to Grammarly and Quillbot for their indispensable roles in refining our manuscript. These tools have significantly enhanced the clarity and conciseness of our writing.
- We acknowledge the use of Adobe Photoshop, Canva, Adobe Illustrator, and Flaticon for creating and enhancing the diagrams and designs in our paper. Their features were crucial in helping illustrate our research findings.
- Lastly, we extend our gratitude to Overleaf for streamlining the process of writing and formatting our paper in LaTeX. Its collaborative features and comprehensive LaTeX environment have greatly facilitated the production of a professional and well-structured document.


## REFERENCES


[1] M. H. Sim, M. Y. Low, C. S. Chong, and M. Shakeri, "Job shop scheduling problem neural network solver with dispatching rules," in *2020 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM).* IEEE, 2020, pp. 514–518.

[2] E. Miller, B. Engelmann, T. Kaupp, and J. Schmitt, "Advanced cascaded scheduling for highly autonomous production cells with material flow and tool lifetime consideration using agvs," *Journal of Machine Engineering*, vol. 23, no. 3, 2023.

[3] E. Miller, T. Kaupp, and J. Schmitt, "Cascaded scheduling for highly autonomous production cells with agvs," in *Global Conference on Sustainable Manufacturing*. Springer, 2022, pp. 383–390.

[4] T. Gabel and M. Riedmiller, "Adaptive reactive job-shop scheduling with reinforcement learning agents," *International Journal of Information Technology and Intelligent Computing*, vol. 24, no. 4, pp. 14–18, 2008.

[5] C. Zhang, W. Song, Z. Cao, J. Zhang, P. S. Tan, and X. Chi, "Learning to dispatch for job shop scheduling via deep reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 1621–1632, 2020.

[6] D. Rooyani, "Efficient two-stage genetic algorithms for comprehensive multi-objective flexible job shop scheduling problems," Ph.D. dissertation, University of Guelph, 2023.

[7] A. Liu, P. B. Luh, K. Sun, M. A. Bragin, and B. Yan, "Integrating machine learning and mathematical optimization for job shop scheduling," *IEEE Transactions on Automation Science and Engineering*, 2023.

[8] E. Morinaga, X. Tang, K. Iwamura, and N. Hirabayashi, "An improved method of job shop scheduling using machine learning and mathematical optimization," *Procedia Computer Science*, vol. 217, pp. 1479–1486, 2023.

# APPENDIX

## A. Project Source Code

The source code for our project, including all the scripts used for Genetic Algorithms and Reinforcement Learning models, is available on GitHub. Interested readers can access, review, and utilize the code under the terms of the specified license. For more details, visit our GitHub repository at: https://github.com/Tharindupriyaharshana/Optimizing-Job-Shop-Scheduling-with-AI