Proceedings of the 2025 Winter Simulation Conference
E. Azar, A. Djanatliev, A. Harper, C. Kogler, V. Ramamohan, A. Anagnostou, and S. J. E. Taylor, eds.

FROM SCENARIO FARMING TO LEARNING: A MODULAR LOW-CODE FRAMEWORK FOR DECISION SUPPORT IN SCHEDULING

Madlene Leißau, and Christoph Laroque

Research Group Industry Analytics, University of Applied Sciences Zwickau, Zwickau, GERMANY

ABSTRACT

Modern manufacturing environments, such as semiconductor manufacturing, require agile, data-driven decision support to cope with increasing system complexity. Discrete event simulation (DES) is a key method for evaluating scheduling strategies under uncertainty. Building on a previously introduced low-code framework for scenario farming, this paper presents an extended and evolving approach that integrates machine learning (ML) to enhance scheduling decision support. The framework automates model generation, distributed experimentation, and systematic scenario data collection, providing the basis for training data-driven decision models. Using the Semiconductor Manufacturing Testbed 2020 as a reference, initial experiments demonstrate how simulation-based insights can be transformed into intelligent scheduling aids. Key features include model structure synchronization across simulation tools, automated experiment design, and modular integration of learning components, providing a first step towards adaptive, simulation-driven decision support systems.

1 INTRODUCTION

Manufacturing environments are undergoing rapid transformation, becoming far more complex, dynamic, and data-driven than in the past. Emerging Industry 5.0 trends and the resulting increased competition are putting traditional approaches (e.g., fixed scheduling rules or mathematical optimization) to the test. As such, they quickly become obsolete when machines fail, orders change, or production conditions change (Tassel et al. 2023; Ngwu et al. 2025). In response, manufacturing companies are increasingly turning to more adaptive approaches. One prominent example is discrete-event simulation (DES), which enables the detailed analysis and optimization of production processes under varying conditions.

As highlighted by Vieira et al. (2018), DES methods and tools are of growing importance as key technologies. DES models allow the consideration and evaluation of different scenarios, e.g., to validate concrete production plans and/or product mixes under the assumption of defined conditions in a risk-free environment without requiring excessive financial or time resources (VDI 3633 2014, Reif et al. 2023).

While DES offers notable advantages, persistent challenges identified in the literature reveal important limitations when it comes to scheduling applications. One major difficulty is accurately representing the system complexity and stochastic uncertainties that characterize modern manufacturing environments. Although DES allows for a level of detail that exceeds traditional analytical methods, this advantage is offset by significant computational requirements. Furthermore, the development and application of DES models in simulation studies or as digital twins (DT) requires a thorough description of the production system, adequate data provision, and efficient data management. In addition, the success of simulation efforts depends heavily on the availability and quality of input data – such as processing times, setup times, and arrival rates – which, despite advances in IoT and MES systems, remains a resource-intensive task in practice (Lugaresi and Matta 2020; Schlecht et al. 2023; Reif et al. 2023).

Beyond these fundamental limitations, other challenges complicate the application of DES in practical scheduling contexts. Navigating the vast solution space of possible sequences and evaluating each candidate through simulation remains computationally demanding, often limiting practical application (Pappert et al.

2023). Compounding these issues is the lack of seamless integration between simulation-driven scheduling and adjacent PPC functions, such as real-time control and execution systems, which can limit the operational impact of simulation studies. Finally, decision support must increasingly balance multiple objectives – including throughput, timeliness, and sustainability – while addressing human and organizational factors that influence the adoption and effective use of DES-based tools.

To overcome these limitations and improve the practical utility of simulation-based planning, recent developments have increasingly focused on three complementary approaches: automating model generation, applying data farming techniques, and using machine learning (ML) for operational decision support. In this context, DES models play a central role by serving as a controlled setting for the systematic execution of large-scale simulation experiments. Through targeted variation of input parameters and operating conditions, commonly referred to as data farming, large amounts of synthetic data can be generated to characterize and predict system behavior over a large solution space (e.g. Sanchez 2021). These large datasets, in turn, provide a valuable foundation for applying ML techniques that can extract actionable insights and predictive patterns (Pappert and Rose 2022). The focus is not on real-time control, but on providing robust, data-driven support for operational decisions, enabling manufacturing systems to adapt more effectively to variability, uncertainty, and multiple competing objectives.

Addressing these demands requires simulation frameworks that automate large-scale experiments and systematically extract actionable knowledge from the data. Previous research (Leißau and Laroque 2024) introduced a modular low-code framework that enables automated scenario farming across simulation environments and synchronization of model structures, as demonstrated with the Semiconductor Manufacturing Testbed 2020 (SMT2020) (Kopp et al. 2020). While this reduced manual setup and execution efforts, results analysis still relied on traditional methods.

This work extends the framework with a learning-based component designed to systematically transform farmed simulation data into intelligent planning decision support. By integrating ML into the scenario farming workflow, the approach aims to lower cognitive and technical barriers, accelerate the generation of actionable insights, and move towards adaptive, simulation-informed scheduling support systems. A first proof-of-concept demonstrates the potential of this extension using SMT2020 as an experimental environment.

The paper is organized as follows: Section 2 reviews the relevant literature on model generation, data farming, and ML approaches for decision support. Section 3 introduces an extended framework, highlighting its modular structure and learning components. Section 4 outlines the test case. Section 5 analyzes the results and Section 6 summarizes the results and discusses future research directions.

2 RELATED WORK

2.1 Model Generation

The automated generation and efficient use of DES models are critical enablers for simulation-based planning and decision support in manufacturing environments. As outlined in VDI 3633 (2014), the development and execution of DES models involves multiple challenges, including data availability, acquisition, management, model creation, validation, and application to specific decision problems.

Recent research emphasizes the importance of automating the model generation process to better manage complexity, variability, and the need for faster simulation-driven insights. Lugaresi and Matta (2020) propose a structured approach that divides model generation into phases such as data acquisition, material flow identification, statistical analysis, rule extraction, code translation, and model validation. Schlecht et al. (2023) extend this view by highlighting the role of model synchronization, adaptation, and dynamic updating, referring to concepts such as generic template-based simulation (GTS), data-driven simulation (DDS), and dynamic data-driven simulation (DDDS) (Haouzi et al. 2013). These paradigms use operational data to automatically generate, update, and refine simulation models in near real time.

Several methods have emerged to support automated DES model generation. For example, process mining techniques are used to derive simulation models from event logs and production data (Lugaresi and

Matta 2021; Zhu et al. 2023). Complementary approaches focus on data-driven modeling structures and adaptive algorithms to translate operational information into executable simulation models (Goodall et al. 2019). The need for automated, robust data processing frameworks to minimize manual effort and reduce errors is widely recognized (Johansson et al. 2003; Vieira et al. 2018). In addition, the use of reusable model building blocks (Reif et al. 2023) can further accelerate model development and adaptation when process structures or operating conditions change.

The semiconductor industry is a particularly relevant application domain for these approaches. Early work (Tulis et al. 1990; Kim et al. 2009; Rank et al. 2015) addressed key manufacturing dynamics but was typically limited to fixed-scope models. More recent studies (Sadeghi et al. 2016; Khemiri et al. 2021) have demonstrated the potential of data-driven fab simulation by integrating elements such as equipment failures, maintenance activities, and transportation logistics. These contributions also highlight persistent challenges related to data adequacy and semantic consistency in the development and deployment of highly automated simulation models.

2.2 Data Farming

The concept of data farming originally emerged from military simulation efforts aimed at improving decision making in dynamic and uncertain environments. Horne and Schwierz (2016) define it as an interdisciplinary methodology that integrates simulation, high-performance computing, and statistical analysis to systematically explore complex systems. According to Sanchez (2021), data farming studies aim to generate large, high-quality simulation datasets that allow for deeper insights into system behavior and the discovery of both expected and unexpected outcomes.

While the origins of data farming are rooted in military applications (e.g., Horne and Schwierz 2008; Horne and Meyer 2010), the approach has increasingly gained relevance in industrial contexts. In the semiconductor industry, for example, Pappert et al. (2017) apply data farming techniques to develop a capacity planning tool that estimates equipment utilization targets under varying production conditions. By using regression models to interpolate missing data, their approach enables fast and robust decision support. Building on this, Pappert and Rose (2022) describe how pre-generated simulation datasets can be used to train neural networks, enabling immediate decision support without the need to repeat time-consuming simulations for each scenario.

These examples demonstrate the potential of data farming to significantly improve the robustness and efficiency of planning processes by providing a richer and more comprehensive database. However, recent studies (Król et al. 2013; Sanchez et al. 2021; Pappert et al. 2023) highlight ongoing challenges: as planning problems become more complex and detailed, the computational demands of large-scale simulation studies also increase, placing new demands on simulation frameworks and decision support architectures.

2.3 Learning Approaches

The use of ML in combination with DES has become an important strategy for improving scheduling decisions in complex manufacturing environments. ML techniques offer the ability to recognize patterns, predict outcomes, and adapt strategies based on large amounts of simulation data, addressing the key challenges of variability and dynamic change (Sarker 2021).

Using learning-based methods for scheduling with simulation has evolved in several different directions. Early approaches primarily used supervised learning techniques to predict performance metrics or replicate scheduling decisions based on simulation data (Bergmann et al. 2015; Murphy et al. 2019; Goodall et al. 2019). In addition to classical supervised learning methods, hybrid approaches such as fuzzy agent-based systems have been used to improve performance prediction in manufacturing simulations and enhance decision support for scheduling tasks (Chen and Wang 2021).

With the increasing availability of computational resources and advances in deep learning, more dynamic strategies have emerged, including the use of deep neural networks trained on simulation data to support rapid scheduling decisions (Pappert and Rose 2022) and ensemble learning methods for adaptive dispatch rule selection (Priore et al. 2018).

Building on these developments, reinforcement learning (RL) has gained particular attention. Advances in deep RL algorithms (e.g., deep Q-networks, actor-critical methods) and increasing computational power have enabled agents to autonomously learn scheduling policies by interacting with detailed DES models (Liu et al. 2022; Sakr et al. 2023). Unlike supervised or ensemble methods, which typically predict outcomes or choose among predefined rules, RL approaches enable the discovery of entirely new strategies through trial-and-error learning, providing a promising path toward fully adaptive, simulation-based scheduling in complex and dynamic environments (Tassel et al. 2023; Yedidsion et al. 2022).

Reflecting these developments, recent research can be broadly categorized into three main streams: supervised and ensemble learning approaches, reinforcement learning approaches, and evolutionary or hybrid approaches. While there is overlap – for example, some reinforcement learning models use supervised components during pretraining or action selection (Tassel et al. 2023; Alexopoulos et al. 2023) – this classification serves as a useful framework for analyzing current approaches and identifying emerging trends in simulation-based scheduling research.

3 SOLUTION APPROACH

Below, the findings of Section 2 are considered in relation to the identified challenges, and an approach is proposed that integrates three main components: a generic simulation model tailored to semiconductor manufacturing, a low-code framework for automated scenario generation and execution, and a learning component aimed at evaluating the feasibility and potential of ML-based decision support.

3.1 Concept

The development and application of DES models in scheduling studies typically requires considerable time, expertise, and technical resources. The creation of simulation models and the execution of experiments – especially in complex domains such as semiconductor manufacturing – are often tightly coupled, manually driven, and difficult to scale. This limits the ability to explore larger solution spaces, generate consistent training data for ML, or compare different scheduling strategies under varying conditions.

In order to address these limitations, the proposed solution approach adopts a modular architecture consisting of three core components. First, a reusable, generic simulation model tailored to semiconductor manufacturing provides a configurable environment for simulation studies. Second, a low-code framework for scenario farming automates the generation and execution of simulation runs, ensuring reproducibility and reducing manual effort. Third, a learning component is integrated to explore the applicability of ML methods within this workflow.

The learning component does not aim to provide fully automated scheduling but rather serves as a demonstrator to assess whether ML models can be effectively trained using structured simulation data. It provides a controlled environment for testing different ML techniques, analyzing their performance, and identifying practical constraints such as data requirements or generalization issues. The overall approach supports methodical experimentation and lays the foundation for future extensions towards adaptive, ML-based scheduling systems.

3.2 Generic Semiconductor Simulation Template

The generic semiconductor simulation template streamlines the creation, validation, and execution of DES models in semiconductor manufacturing. It is implemented in AnyLogic (https://www.anylogic.com/) and relies on generic building blocks and reusable components that provide a flexible basis for rapidly generating accurate simulation models across different production scenarios. In this section, we outline the core elements of the template, detailing its structure, methodology, and intended applications.

3.2.1 Objectives and Scope

As part of the modular architecture described in Section 3.1, the template is intended to facilitate efficient model generation for semiconductor manufacturing processes, thereby reducing the manual effort typically involved in building custom DES models. The template is designed to address common model creation challenges such as data integration, standardization, and adaptability. By providing a flexible yet standardized approach, the template helps to:

- minimize model development time and reduce the need for extensive programming skills
- ensure consistency across different simulation studies, leading to improved model reusability and comparability
- enable non-technical stakeholders to participate in the model development process, fostering collaboration across interdisciplinary teams

The simulation template is designed to be applicable to various areas of semiconductor manufacturing, including production planning, bottleneck analysis, and capacity assessment. The modular structure allows users to adapt the template to the specific needs of different production lines or factory environments.

3.2.2 Implementation

The modeling logic is based on a two-level agent structure implemented in AnyLogic. At the top level, tool group agents represent groups of functionally similar machines, while tool agents within each group model individual production tools. This hierarchical structure supports modularity, simplifies initialization, and reduces the need to manually model each tool or process step.

Production lots are released into the model according to a configurable schedule and assigned a product type, which determines the processing recipe. Each recipe contains a sequence of steps, including processing times, sampling probabilities, and tooling requirements. The quantity of wafers per lot is defined by a dedicated parameter (wafer size), ensuring flexible representation of lot characteristics.

For each step, the model identifies the appropriate toolgroup by matching the characteristics specified in the recipe with the available toolgroups in the simulation. Once a toolgroup is selected, operations such as dispatching, sampling, batching, and rework handling are performed within the toolgroup agent. This centralization at the group level allows for efficient coordination and consistent logic across tools.

Batching is handled proactively to reduce wait times, with lots grouped based on configurable criteria (e.g., product type, process step). After batching, the tool selection process considers tool availability and constraints such as lot-to-lens dedication or setup requirements. Capacity constraints are applied per tool and configured via cascading parameters during model initialization.

Stochastic tool failures and preventive maintenance are implemented as separate mechanisms. Failures are modeled explicitly using custom logic to reflect random failure events, while preventive maintenance follows independently defined schedules, supporting both time-based and usage-based strategies.

This agent-based modeling approach provides a flexible yet structured way to represent complex semiconductor processes, enabling scalable and reusable simulation logic across different scenarios.

3.3 Low-Code Framework For Scenario Farming

Efficient experimentation with simulation models – especially when exploring large solution spaces or training ML models – requires a high degree of automation, consistency, and repeatability. Manual setup and execution of simulation runs is not only time-consuming, but also error-prone, making it unsuitable for data-intensive applications such as data farming or ML-based decision support. To address these challenges, a low-code framework has been developed that enables automated scenario generation, execution, and data collection across multiple simulation environments.

3.3.1 Objectives and Scope

The primary goal of the low-code scenario farming framework is to enable efficient and scalable experimentation with DES models by automating the generation, execution, and evaluation of simulation scenarios. Unlike traditional simulation studies, which often rely on manual configuration and execution, the framework is designed to support structured experimentation workflows, reduce technical barriers, and promote reproducibility. Specifically, the framework aims to:

- Reduce the manual effort and complexity of setting up and running experiments, especially in data-intensive applications such as robust testing or ML.
- Enable non-experts to define and run simulation scenarios through a low-code interface, facilitating broader participation across interdisciplinary teams.
- Seamlessly integrate with parameterized simulation models, such as the generic semiconductor model introduced in Section 3.2, by exposing configurable input parameters and supporting standardized output formats.
- Ensure consistent and reproducible execution across simulation runs, enabling comparative analysis and systematic exploration of solution spaces (e.g., data farming).
- Support downstream integration with learning pipelines, allowing simulation-generated data to be used directly for training and validation of ML models. This aspect is explored in more detail in Section 3.4, where the learning component is introduced as a proof-of-concept for ML integration within the framework.

The framework is domain-agnostic in principle, but it has been tailored and tested in the context of semiconductor manufacturing, where large-scale scenarios and reproducible evaluations are particularly critical. Its modular structure allows it to be adapted to other use cases that require systematic simulation-based experimentation, such as capacity planning, scheduling rule comparison, or sensitivity analysis.

3.3.2 Implementation

The scenario farming framework is implemented in KNIME, an open-source, node-based analytics platform that supports low-code development and visual workflows. This platform was chosen to ensure ease of use, transparent data handling, and straightforward extensibility for future use cases.

The current implementation includes the following core submodules (as of this writing):

1. *Import of a DES model*: This module allows users to load a DES model, including its file structure, simulation entry point, and optional configurations such as database connections or parameter files. This supports model reuse across experiments and ensures a standardized setup process.
2. *Model preview and validation*: Once imported, the simulation model is parsed, and its internal structure is displayed. Input tables, configurable parameters and variables are visualized in table views to aid model verification and preparation. This step is critical to ensure consistency and avoid downstream configuration errors.
3. *Factor configuration*: In this step, the user selects the input parameters (factors) to be explored in the solution space. For each factor, the user can define the minimum and maximum values, as well as a step size, thereby specifying the resolution and boundaries of the parameter variation. These settings determine the granularity of the experiment and define the base configuration for subsequent scenario generation. The factor definitions are stored internally and passed to the next step for use in the experiment design process.
4. *Design of Experiments (DoE)*: Based on the factor definitions from the previous step, users can configure key parameters of the simulation study-such as simulation start time, run length, number of replications, and random seed control. In addition, a DoE method can be selected (e.g., full factorial, Latin-Hypercube Sampling (LHS)), which is then used to generate a structured set of

experimental configurations. The resulting design is validated and written to the attached experiment database, forming the executable scenario matrix for distributed simulation.

5. *Distributed Execution*: The simulation runs are executed across local or distributed environments to enable scalable data generation. The infrastructure is based on Docker, allowing simulation models to be containerized and executed in isolated, reproducible environments. Within KNIME, Python scripts are used to automate key container management tasks: writing Docker images, creating and starting containers with scenario-specific configurations, monitoring their execution status, and automatically removing containers once simulations are complete. This architecture supports parallel and flexible deployment, significantly reduces experiment runtime, and enables efficient execution of large-scale simulation studies in a controlled and reproducible manner.
6. *ML training and validation*: After execution, the simulation output is collected, pre-processed, and prepared for downstream ML tasks. This includes feature extraction, data cleaning, and goal definition. The processed datasets can then be used to train, validate, and compare ML models. This integration aspect is discussed in more detail in Section 3.4.

All submodules are implemented as reusable KNIME components and integrated into a unified workflow. The design emphasizes modularity, transparency, and extensibility, allowing users to modify or extend individual components without changing the entire workflow.

3.4 Learning Component

The final module of the proposed solution architecture is a learning component that explores the applicability of ML techniques in simulation-based scheduling. In its current form, the component serves as a proof-of-concept within the scenario farming workflow introduced in Section 3.3. It demonstrates how simulation-generated data can be structured, used for model training, and evaluated to support data-driven analysis and decision support. While not yet intended as an operational scheduling tool, the component is designed with extensibility in mind and can form the basis for future integration into adaptive, ML-supported scheduling systems.

3.4.1 Objectives and Scope

The learning component is designed to investigate how ML methods can be systematically applied to support scheduling decisions based on simulation-generated data. It aims to explore the conditions under which models can be trained to generalize from large-scale scenario data and provide predictive or prescriptive insights into production system behavior. The focus is on demonstrating feasibility and identifying methodological requirements, rather than delivering a deployable scheduling solution.

The component is embedded in the KNIME-based workflow and closely linked to the scenario farming framework described in Section 3.3. This setup allows automated data preparation, feature extraction, model training, and performance evaluation under reproducible conditions. It also allows researchers and practitioners to test different ML approaches - such as decision trees, ensemble models, or reinforcement learning agents - without requiring changes to the underlying simulation model.

While the current scope is exploratory, the design is intentionally open for extension. In the future, the component may evolve into a modular learning layer that interacts with real-time scheduling systems or digital twins to support adaptive decision making in complex manufacturing environments.

3.4.2 Implementation

The learning component is technically embedded in the KNIME-based scenario farming workflow and is designed to operate on structured simulation output generated by the execution module. It provides a modular environment for preparing data, defining learning problems, training machine learning models, and evaluating their performance, all within a reproducible and extensible workflow.

The implementation follows a typical supervised learning pipeline: data transformation, feature engineering, dataset partitioning, model training, and evaluation. Simulation results are collected from the experiment database and preprocessed to convert timestamps, durations, identifiers, and categorical inputs into numerical formats suitable for model training. Derived features and domain-specific transformations can be incorporated as needed, depending on the target use case (e.g., scheduling, resource planning, or performance prediction). The component supports integration with Python-based machine learning libraries and includes optional use of KNIME's AutoML capabilities, which allow automatic algorithm selection, feature filtering, hyperparameter tuning, and validation. These functionalities are accessible through low-code interfaces and are designed to lower the barrier for non-expert users to experiment with different learning approaches. Common evaluation metrics such as accuracy, mean absolute error (MAE), or area under the ROC curve (AUC) are used to assess model performance depending on the selected problem type.

The architecture is intentionally designed to be use case agnostic. Whether the learning goal is binary classification, regression, or even policy approximation, the component provides a structured interface for applying and comparing different ML methods to simulation-derived datasets. This flexibility supports iterative experimentation and lays the foundation for integrating learning-based logic into future adaptive scheduling or planning systems.

4 TEST CASE: SEMICONDUCTOR MANUFACTURING TESTBED2020

The SMT2020 includes four simulation models, each designed to address different complexities in semiconductor manufacturing. Of particular interest for this study is dataset 2, which represents a low-volume, high-mix (LVHM) environment. This model is tailored to simulate the operational intricacies of modern fabs that handle multiple product types under tight delivery schedules.

The LVHM model features 10 product types with process routes ranging from 242 for type 5 to 583 steps for type 3, capturing the high variability and customization typical of LVHM factories. The inclusion of advanced features such as re-entrant flows, critical queue time (CQT) constraints, and cascading tool operations reflects the operational complexity inherent in this manufacturing environment.

The toolset in the LVHM model includes 105 toolgroups with a number of 913 tools organized into functional areas such as diffusion, implant, lithography, and metrology. Specific operational constraints, such as lot-to-lens dedication and customized maintenance schedules, highlight the precision required in these manufacturing processes. The model also accounts for hot lots, which require dynamic prioritization and flexible setup strategies to accommodate sudden changes in production requirements.

This LVHM model serves as a critical testbed for evaluating advanced operational strategies, providing researchers and practitioners with a robust platform to explore and refine production control mechanisms in semiconductor manufacturing. A detailed description of the models for the entire testbed can be found in the literature (Hassoun et al. 2019; Koop et al. 2020a; Kopp et al. 2020b). In addition, the model data can be downloaded from https://p2schedgen.fernuni-hagen.de/.

5 EXPERIMENT

Based on the solution approach as a proof-of-concept, the test case described in the previous section is now considered within the framework in terms of the execution of large-scale simulation experiments and applicability of ML methods.

5.1 Objectives and Definition of the Simulation Experiment

We aimed to achieve two objectives with the simulation experiment. First, we aimed to generate a large, diverse set of synthetic production data to serve as a training and validation base for ML models within the proposed framework. Second, we wanted to systematically vary tool availability in a selected process area to generate simulation data that reflects realistic scheduling dynamics under resource constraints.

For this purpose, we focused on a key resource parameter in the generic semiconductor simulation model: the number of tools in all toolgroups assigned to the dielectric area in the SMT2020 2dataset. For each of these toolgroups, the number of tools was sampled independently between a minimum of 2 tools and the original number of tools defined in the dataset, using steps of 1. The resulting parameter space was explored using LHS to ensure a statistically efficient and well-distributed set of configurations. A total of 5000 unique simulation scenarios were generated.

Each configuration was simulated over a 150-day time horizon to generate rich and consistent process data. The output includes detailed lot level information such as processing sequences, tool assignments, completion times, and associated metadata.

The resulting data was not directly evaluated in the simulation study itself but was prepared for downstream ML tasks. ML models were later trained to predict whether individual lots would meet their originally defined due dates under different resource configurations. This evaluation of schedule adherence was not part of the simulation analysis but formed the central prediction task for the learning component described in Section 3.4.

5.2 Results

All configurations were automatically started in Docker containers on a large virtual machine with Ubuntu 22.04 LTS as operating system on a dedicated simulation server of the University of the Bundeswehr Munich. An Intel Xeon Platinum 8362 with 2.80 GHz is installed as the CPU in the simulation server. The virtual machine is assigned 128 cores and a total of 1 TB RAM and 1 TB hard drive storage.

As part of the simulation experiment, 50 Docker containers were started in parallel. For 5,000 configurations and a real duration of around 7 minutes for each simulation run, this meant a total processing time of ~12 hours. By way of illustration, it would take approximately 24 days to run the configurations sequentially – assuming they can be started immediately one after the other.

After running the simulation and preprocessing the data, the resulting dataset was used to train and evaluate several supervised ML models using KNIME's AutoML framework. The prediction task focused on classifying production lots as on time or late based on their completion date relative to the original due date (±24 hours tolerance).

Among the models tested, Gradient Boosted Trees and Random Forests achieved the best overall performance. Gradient Boosted Trees achieved the highest values for AUC (~0.99), Accuracy (~0.95), and F-Measure (~0.91), indicating excellent classification ability and robustness even under class imbalance. Naïve Bayes also performed well, especially in terms of Balanced Accuracy, suggesting good sensitivity to both classes. In contrast, simpler models such as Random Forest and Logistic Regression showed limitations, particularly in terms of F-measure and Balanced Accuracy-reflecting challenges in correctly identifying late batches. The ROC curves further support these findings, with ensemble-based models (Gradient Boosted Trees, Random Forest) showing steep and well-separated curves, while linear models lag, see Figure 1.

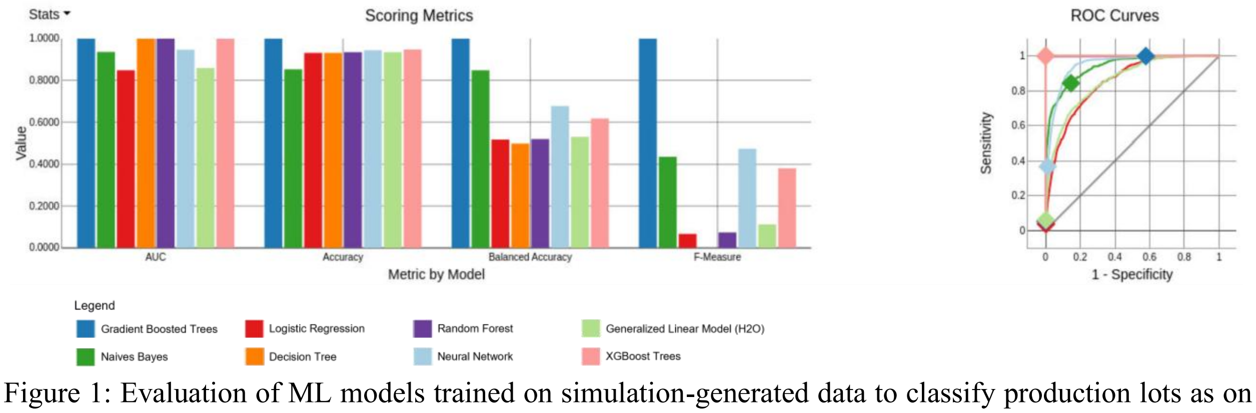

Figure 1: Evaluation of ML models trained on simulation-generated data to classify production lots as on time or late. The left chart compares models across common performance metrics (AUC, Accuracy, Balanced Accuracy, F-Measure), while the right chart shows corresponding ROC curves.

Overall, the results demonstrate that ML models can learn meaningful patterns from the simulation-generated dataset, enabling reliable prediction of on-time delivery based on the production context. These results validate the feasibility of the proposed framework and highlight the potential of integrating simulation and learning for scheduling decision support.

5.3 Challenges and Limitations

Despite the encouraging results, there are still several limitations. The approach relies on synthetic data from simulation models, which may not fully capture variability in the real world. Consequently, trained ML models may encounter generalization issues when applied to unseen or atypical scenarios.

Additionally, rare events and nonlinear dynamics may be underrepresented in the generated datasets, which limits the predictive accuracy for edge cases. While the current focus has been on supervised learning for binary classification, extending to more complex tasks (e.g. reinforcement learning or multi-objective settings) introduce further methodological and computational challenges.

Further work is also required to operationalize integration: translating ML outputs into actionable scheduling decisions requires technical interfaces, organizational acceptance, and trust in the system's recommendations. Advancing the framework towards an operational decision support system requires closer integration with existing planning processes and the seamless embedding of learning-based components in day-to-day decision-making workflows.

6 CONCLUSION AND OUTLOOK

Simulation experiments take time and effort. There is little difference between whether a model must be created or already exists. Even if an existing model only needs to be updated with new parameters, the simulation itself takes time to run (Pappert and Rose 2022). This becomes a practical limitation when simulation is used not only for evaluation, but also as a basis for training ML models that require large and diverse datasets. The framework presented addresses this challenge by integrating a reusable simulation template, automated scenario farming, and a learning component. Together, these elements enable scalable experimentation and structured data generation for decision support for scheduling.

In this context, the experimental results confirm that meaningful patterns can be extracted from simulation-generated data to predict schedule adherence. This validates the feasibility of using simulations to support operational decision-making.

As a next step, the authors will increase the functional depth of the simulation template itself. Planned enhancements include the implementation of additional control mechanisms (e.g., dynamic scheduling rules, more complex batching strategies, lot splitting) and the refinement of process constraints. These additions are intended to improve the variability and expressiveness of simulation scenarios, thereby enriching the factor space available for analysis and learning.

In addition, a backward simulation approach, starting from desired target states, is explored as a basis for planning under delivery constraints. To improve transparency and trust in the trained models, explainable AI techniques will be integrated into the workflow. In parallel, transfer learning strategies will be evaluated to reduce training effort and improve adaptability across simulation contexts.

To promote reusability and collaboration, the entire framework – including the low-code workflow and simulation template – will be published on GitHub. This will support further development, transparency, and adoption by the simulation and operations research communities.

Taken together, the proposed framework provides a flexible foundation for developing simulation-driven, learning-enhanced scheduling tools. The authors intend to gradually evolve this approach into a modular decision support system to assist planners in complex, uncertain and time-constrained production environments.

ACKNOWLEDGMENTS

We want to thank Professor Oliver Rose and his team for the provision of a virtual machine on their simulation cluster.

This work was partially funded by the Future Mobility project. The Future Mobility project is co-funded by the European Regional Development Fund (ERDF) and by tax revenues based on the budget approved by the Saxon State Parliament.

Co-funded by the European Union

Freistaat SACHSEN This project is co-financed from tax revenues on the basis of the budget adopted by the Saxon State Parliament.

REFERENCES

Alexopoulos, K., Mavrothalassitis, P., Bakopoulos, E., Nikolakis, N., and D. Mourtzis. 2025. "Deep Reinforcement Learning for Selection of Dispatch Rules for Scheduling of Production Systems". In *Applied Sciences*, 15(1):232.

Bergmann, S., Feldkamp, N., and S. Straßburger. 2015. "Approximation of Dispatching Rules for Manufacturing Simulation using Data Mining Methods". In *2015 Winter Simulation Conference (WSC)*, 2329–2340 http://dx.doi.org/10.1109/WSC.2015.7408344.

Chen, T.-C. T., and Y.-C. Wang. 2021. "Fuzzy Dynamic-Prioritization Agent-Based System for Forecasting Job Cycle Time in a Wafer Fabrication Plant". In *Complex & Intelligent Systems* 7: 2141-2154.

Goodall, P. A., Sharpe, R., and A. West. 2019. "A Data-Driven Simulation to Support Remanufacturing operations". In *Computers in Industry (105)*:48–60.

Haouzi, H., Thomas, A., and P. Charpentier. 2013. "Toward Adaptive Modelling & Simulation for IMS: The Adaptive Capability Maturity Model and Future Challenges. In *11th IFAC Workshop on Intelligent Manufacturing Systems (IMS'2013)*, 22th–25th May, Brazil, São Paulo, 174–179.

Horne, G. E., and K.-P. Schwierz. 2008. "Data Farming around the World Overview". In *2008 Winter Simulation Conference (WSC)*, 1442–1447 https://doi.org/10.1109/WSC.2008.4736222.

Horne, G. E., and T. E. Meyer. 2010. "Data Farming and Defense Application". In *MODSIM World Conference and Expo*., 13th October, USA, Hampton (VA).

Horne, G. E., and K.-P. Schwierz. 2016. "Summary of Data Farming". In *Axioms*, 5(1) 8 http://dx.doi.org/10.3390/axioms5010008.

Johansson, B., Johnsson, J., and A. Kinnander. 2003. "Information Structure to Support Discrete Event Simulation in Manufacturing Systems". In *2003 Winter Simulation Conference (WSC)*, 1290–1295 https://doi.org/10.1109/WSC.2003.1261564.

Khemiri, A., Yugma, C., and S. Dauzère-Pérès. 2021. "Towards a Generic Semiconductor Manufacturing Simulation Model". In *2021 Winter Simulation Conference (WSC)* https://doi.org/10.1109/WSC52266.2021.9715349.

Kim, B.-I., Jeong, S., Shin, J., Koo, J., Chae, J., and S. Lee. 2009. "A Layout- and Data-Driven Generic Simulation Model for Semiconductor Fabs". In *IEEE Transactions on Semiconductor Manufacturing* 22(2): 225–231.

Kopp, D., M. Hassoun, A. Kalir, and L. Mönch. 2020a. "SMT2020 – A Semiconductor Manufacturing Testbed". In *IEEE Transactions on Semiconductor Manufacturing* 33(4):522–531.

Kopp, D., M. Hassoun, A. Kalir, and L. Mönch. 2020b. "Integrating Critical Queue Time Constraints into SMT2020 Simulation Models". In *2020 Winter Simulation Conference (WSC),* 1813–1824, https://doi.org/10.1109/WSC48552.2020.9383889.

Król, D., Wrzeszcz, M., Kryza, B., Dutka, L, and J. Kitowski. 2013. "Massively Scalable Platform for Data Farming Supporting Heterogeneous Infrastructure". In *4th International Conference on Cloud Computing, Grids, and Virtualization*, 27th May – 1st June, Spain, Valencia, 144–149.

Leißau, M., and C. Laroque. 2024. "Breaking Barriers in Semiconductor Simulations: An Automated Low-Code Framework for Model-Structure Synchronisation and Large-Scale Simulation Studies". In *2024 Winter Simulation Conference (WSC)*, 1919–1930, https://doi.org/10.1109/WSC63780.2024.10838804.

Liu, J., Qiao, F., Zou, M., Zinn, J., Ma, Y., and B. Vogel-Heuser. 2022. „Dynamic Scheduling for Semiconductor Manufacturing Systems with Uncertainties Using Convolutional Neural Networks and Reinforcement Learning". In *Complex & Intelligent Systems* 8:4641–4662.

Lugaresi, G., and A. Matta. 2020. "Generation and Tuning of Discrete Event Simulation Models for Manufacturing Applications". In *2020 Winter Simulation Conference (WSC)* 2707–2718, http://dx.doi.org/10.1109/WSC48552.2020.9383870.

Lugaresi, G., and A. Matta. 2021. "Discovery and Digital Model Generation for Manufacturing Systems with Assembly Operations". In *17th International Conference on Automation Science and Engineering (CASE)*, 23th–27th August, Lyon, 752–757.

Murphy, R., Newell, A., Hargaden, V., and N. Papakostas. 2019. "Machine Learning Technologies for Order Flowtime Estimation in Manufacturing systems.". In *52nd CIRP Conference on Manufacturing Systems (CMS)*, 12th–14th June, Slovenia, Ljubljana, 701–706.

Ngwu, C., Liu, Y., and R. Wu. 2025. "Reinforcement Learning in Dynamic Job Shop Scheduling: A Comprehensive Review of AI-Driven Approaches in Modern Manufacturing". *Journal of Intelligent Manufacturing* https://doi.org/10.1007/s10845-025-02585-6.

Pappert, F. S., Rose, O., and F. Suhrke. 2017. "Simulation Based Approach to Calculate Utilization Limits in Opto Semiconductor Frontends". In *2017 Winter Simulation Conference (WSC)*, 3888–3898 https://doi.org/10.1109/WSC.2017.8248099.

Pappert, F. S., and O. Rose. 2022. "Using Data Farming and Machine Learning to Reduce Response Time for the User". In *2022 Winter Simulation Conference (WSC)*, 1707–1718 https://doi.org/10.1109/WSC57314.2022.10015466.

Pappert, F. S., Seufferth, D., Stein, H., and O. Rose. 2023. "Using Kubernetes to Improve Data Farming Capabilities". In *2023 Winter Simulation Conference (WSC)*, 1919–1930 https://dl.acm.org/doi/10.5555/3643142.3643312.

Priore, P., Ponte, B., Puente, J., and A. Gómez. 2018. "Learning-Based Scheduling of Flexible Manufacturing Systems Using Ensemble Methods". In *Computer & Industrial Engineering* 126: 282–291, https://doi.org/10.1016/j.cie.2018.09.034.

Rank, S., Hammel, C., Schmidt, T., and G. Schneider. 2015. "Reducing Simulation Model Complexity by Using an Adjustable Base Model for Path-Based Automated Material Handling Systems - A Case Study in the Semiconductor Industry". In *2015 Winter Simulation Conference (WSC)*, 2896–2907 https://doi.org/10.1109/WSC.2015.7408393.

Reif, J., Jeleniewski, T., and A. Fay. 2023. "An Approach to Automating the Generation of Process Simulation Sequences". In *28th International Conference on Emerging Technologies and Factory Automation (ETFA)*, 12th–15th September, Sinaia.

Sadeghi, R., Dauzere-Pérès, S., and C. Yugma. 2016. "A Multi-Method Simulation Modelling for Semiconductor Manufacturing". In *19th World Congress of the International Federation of Automatic Control*, 28th–30th June 2016, Troyes, 727–732.

Sakr, A. H., Aboelhassan, A., Yacout, S., and S. Bassetto. 2023. "Simulation and Deep Reinforcement Learning for Adaptive Dispatching in Semiconductor Manufacturing Systems". In *Journal of Intelligent Manufacturing* 34:1311–1324.

Sanchez, S. M., Sanchez, P. J., and H. Wan. 2021. "Work Smarter, Not Harder: A Tutorial on Designing and Conducting Simulation Experiments". In *2021 Winter Simulation Conference (WSC)*, http://dx.doi.org/10.1109/WSC52266.2021.9715422.

Sanchez, S. M. 2021. "Data Farming: The meanings and methods behind the metaphor". In *Proceedings of the Operational Research Society Simulation Workshop 2021 (SW21)*, 22th– 24th March, UK, Loughborough, 10–17.

Sarker, I. H. 2021. "Machine Learning: Algorithms, Real-World Applications and Research Directions". In *SN Computer Science, SCI.* 2:160 https://doi.org/10.1007/s42979-021-00592-x.

Schlecht, M., De Guio, R., and J. Köbler. 2023. "Automated generation of simulation model in context of industry 4.0". In *International Journal of Modelling and Simulation*, 441–453, https://doi.org/10.1080/02286203.2023.2206075.

Tassel, P., Kovács, B., Gebser, M., Schekotihin, K., Stöckermann, P., and G. Seidel. 2023. „Semiconductor Fab Scheduling with Self-Supervised and Reinforcement Learning". In *2023 Winter Simulation Conference (WSC)*, 1924–1935 https://doi.org/10.1109/WSC60868.2023.10407747.

Tulis, B., Mehrotra, V., and D. Zuanich. 1990. "Successful Modeling Of A Semiconductor R&D Facility". In *IEEE/SEMI International Symposium on Semiconductor Manufacturing Science*, 21th–23th May, USA, Burlingame, 26–32, https://doi.org/10.1109/ISMSS.1990.66131.

VDI 3633. 2014. *Simulation of systems in materials handling, logistics and production – Fundamentals*. Part 1, Düsseldorf: Beuth.

Vieira, A. A. C., Dias, L. M. S., Santos, M. Y., Pereira, G. A. B., and J. A. Oliveira. 2018. "Setting an Industry 4.0 Research and Development Agenda for Simulation – A Literature Review". In *International Journal of Simulation Modelling*, http://dx.doi.org/10.2507/IJSIMM17(3)429.

Yedidsion, H., Dawadi, P., Norman, D., and E. Zarifoglu. 2022. "Deep Reinforcement Learning for Queue-Time Management in Semiconductor Manufacturing". In *2022 Winter Simulation Conference (WSC)*, 377–390 https://doi.org/10.1109/WSC57314.2022.10015463.

Zhu, L., Lugaresi, G., and A. Matta. 2023. "Automated Generation of Digital Models for Production Lines Through State Reconstruction". In *19th IEEE International Conference on Automation Science and Engineering*, 26th–30th August, Auckland, https://dx.doi.org/10.2139/ssrn.4605293.

AUTHOR BIOGRAPHIES

MADLENE LEIßAU is a research assistant at the Institute of Management and Information at the University of Applied Sciences Zwickau, Germany and works in the research group Industry Analytics (www.industry-analytics.de) in the project Future Mobility. She holds a M.Sc, in management. Her main interest is the application of simulation and machine learning methods in relation to operational decision support for manufacturing systems. Her email address is Madlene.Leissau@fh-zwickau.de.

CHRISTOPH LAROQUE is a full Professor of Business Analytics at the University of Applied Sciences Zwickau, Germany. His research group Industry Analytics (www.industry-analytics.de) is working on the development of manufacturing companies towards data-driven organizations. One focus is the application of simulation-based decision support techniques in manufacutring operations. His email address is Christoph.Laroque@fh-zwickau.de.