Proc. of the European Safety and Reliability Conference (ESREL 2026)
Edited by José C. Matos, Paulo B. Lourenço, Michael Beer, Daniel Oliveira,
Edoardo Patelli, Hélder S. Sousa, Rui A. Silva, Monica Santamaria and Chiara Turco
© 2026 ESREL 2026 Organizers. *Published by* Research Publishing, Singapore.
ISBN: 978-981-94-6718-1 | doi: 10.3850/ESREL2026061419_esrel26-p26109-cd

Integrating Risk into Skill-Based Manufacturing: A Modular Platform for Multi-Objective Optimization

Max Hossfeld

InnovationCampus Future Mobility (ICM), University of Stuttgart, Germany.
E-mail: max.hossfeld@icm.uni-stuttgart.de

Patrick Fischer

Institute of Industrial Automation and Software Engineering (IAS), University of Stuttgart, Germany.
E-mail: patrick.fischer.research@proton.me

Alexander Uhl

Institute for Control Engineering of Machine Tools and Manufacturing Units (ISW), University of Stuttgart, Germany. E-mail: alexander.uhl@isw.uni-stuttgart.de

Andreas Wortmann

Institute for Control Engineering of Machine Tools and Manufacturing Units (ISW), University of Stuttgart, Germany. E-mail: andreas.wortmann@isw.uni-stuttgart.de

Andrey Morozov

Institute of Industrial Automation and Software Engineering (IAS), University of Stuttgart, Germany.
E-mail: andrey.morozov@ias.uni-stuttgart.de

Modern manufacturing systems are increasingly characterized by reconfigurable machinery, volatile production demands, and exposure to operational disturbances such as delays, equipment degradation, and failures. While optimization- and AI-based production planning methods have advanced significantly, there is still a lack of integrated and reproducible platforms for modeling, comparing, and benchmarking these approaches under realistic, risk-informed manufacturing conditions. This paper presents a modular software platform for multi-criteria, risk-informed production planning in skill-based manufacturing systems. The platform models a factory as a composition of machines, skills, and products, represented through modular classes that capture executable capabilities and product transformations. A state–transition formulation maps the current factory state to feasible skill executions, enabling the generation of valid production plans that transform an initial state into a desired product configuration while respecting physical and logical constraints. Each skill execution is associated with execution time, energy consumption, and reliability attributes. This unified representation supports the joint optimization of competing objectives, including minimizing energy consumption and makespan, maximizing reliability, and reducing overall production risk.

The platform enables systematic benchmarking of diverse planning and optimization strategies, ranging from classical methods (e.g., A*, mixed-integer linear programming, constraint programming, and genetic algorithms) to AI-based approaches (e.g., reinforcement learning, symbolic planning, and graph neural networks). A common modeling and execution interface ensures consistent problem definitions and reproducible performance comparisons across algorithms and scenarios. Key contributions include: (i) a machine–skill–product meta-model linking symbolic process descriptions with executable planning, (ii) a state-aware action generator that enforces physical and logical feasibility, and (iii) a benchmarking environment for analyzing trade-offs between competing objectives. The proposed platform provides a foundation for integrating risk and reliability into intelligent production planning, thereby advancing safer, more efficient, and more trustworthy manufacturing systems. The platform is open-source and available via a public Git repository.

Keywords: Risk-informed production planning, Skill-based manufacturing, AI-based scheduling, Multi-objective optimization, Reliability in manufacturing.

1. Introduction

The increasing adoption of skill-based and reconfigurable manufacturing systems has created a demand for flexible scheduling approaches that can handle dynamic configurations and multiple, often conflicting objectives (cf. Froschauer et al. (2022); Pfrommer et al. (2015); Hossfeld and Wortmann (2024)). From a conceptual perspective, such conflicts arise because production performance, efficiency, and dependability do not constitute independent dimensions, but are tightly interwoven in the emergent form of any industrial production system (cf. Hossfeld et al. (2026)).

While a broad range of exact, heuristic, and AI-based scheduling methods has been proposed, their evaluation is typically confined to specific simulation environments or problem formulations. This limits reproducibility and hampers systematic comparison across methods, particularly when criteria such as time, energy consumption, and reliability must be considered jointly.

To address this challenge, this paper presents a modular software platform that separates factory simulation, state and action abstraction, and scheduling logic. By introducing a unified interface between the simulation and scheduling layers, heterogeneous optimization methods can be applied interchangeably to the same manufacturing scenario, enabling consistent benchmarking under identical conditions.

Contributions: The main contributions of this paper are as follows: (i) a machine–skill–product meta-model that links symbolic process descriptions with executable production planning, (ii) a state-aware action generation mechanism that enforces physical and logical feasibility of actions, and (iii) a benchmarking environment that enables the systematic analysis of trade-offs between time, energy consumption, and reliability across different scheduling approaches.

The remainder of this paper is as follows: Section 2 reviews related work. Section 3 introduces the proposed modular platform and its architecture. A demonstration of the platform is presented in Section 4, followed by results and discussion in Section 5. Section 6 concludes the paper.

2. State of the Art

Multi-objective scheduling has been widely studied in manufacturing, most commonly optimizing criteria such as makespan, cost, and energy consumption. Quan et al. (2022) propose a multi-objective scheduling approach based on virtual workflow modeling and evolutionary optimization to handle nonlinear manufacturing processes and reprocessing events, emphasizing dynamic adjustment while optimizing objectives such as makespan and energy consumption. Zhang et al. (2023) address an energy-efficient multi-objective integrated process planning and scheduling (IPPS) problem in remanufacturing, integrating disassembly, flexible job-shop-style reprocessing, and reassembly, and optimizing objectives including completion time and energy-related measures. Rastgar et al. (2023) extend the scope toward joint decision-making by integrating production planning, scheduling, and (imperfect) maintenance activities in a multi-objective framework, demonstrating how reliability-related aspects can be incorporated through maintenance modeling. Beyond manufacturing-specific models, tri-criteria scheduling under energy consumption, time, and reliability constraints has been studied in computing and embedded/workflow settings. Aupy et al. (2012) formalize energy-aware scheduling under makespan and reliability constraints, highlighting the trade-off induced by DVFS and proposing heuristics that improve reliability via task re-execution while controlling energy. Similarly, Módos et al. (2017) study robust production scheduling under energy consumption limits, explicitly accounting for uncertainty (e.g., operation delays) and proposing exact and heuristic algorithms that proactively ensure compliance with energy constraints. Recent work increasingly emphasizes comparative evaluation of optimizers and learning-based methods. Burmeister et al. (2025) provide a comparative analysis of evolutionary algorithms for energy-aware production scheduling, helping to clarify empirical strengths and weaknesses across multi-objective evolutionary variants. Complementing algorithmic studies, Meilanitasari and Shin (2021) survey prediction and op-

timization for sequence-driven scheduling in flexible job shops, including AI/ML-based methods, and highlight trends toward data-driven and hybrid optimization pipelines. Recent work considers reliability and risk at the level of skills and action execution, using both model-based reliability assessment Grimmeisen et al. (2022, 2025).

Overall, existing work either focuses on specific optimization formulations (often energy–time trade-offs), incorporates reliability or robustness in a method-specific manner, or surveys algorithms without providing a unified executable environment for reproducible comparison. In particular, few approaches offer an integrated, modular pipeline that formalizes manufacturing as a machine–skill–product model, generates state-dependent feasible actions consistently, and enables benchmarking of heterogeneous optimization methods while exposing reliability and risk as first-class quantities. The platform proposed in this paper addresses this gap by decoupling simulation and scheduling through a common interface, enabling systematic and reproducible benchmarking across algorithms.

3. Methodology

3.1. *Overview*

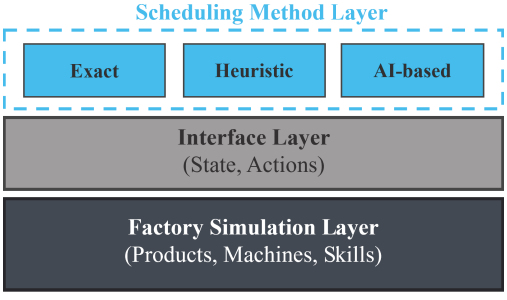


Fig. 1. Overview of the three abstract layers of the modular platform for multi-objective optimization of skill-based manufacturing.

The modular platform for multi-objective optimization of skill-based manufacturing presented in this paper consists of three abstract layers as shown in Fig. 1. The first layer is a simplified factory simulation that includes processing machines, storage machines, and machines used for transporting products between them. Within this layer, products move between machines and are processed according to predefined tasks. The second layer is an interface layer that abstracts the details of the simulation layer for the scheduling layer. It encodes the current state of the factory, including the individual states of all machines and products, and provides a catalog of all feasible actions given the current state. The third and final layer is the scheduling layer, which consists of a set of modular and interchangeable algorithms and optimization strategies. The scheduling methods may be exact, heuristic, or AI-based. Examples of classical methods include A*, mixed-integer linear programming, constraint programming, and genetic algorithms, while AI-based approaches include reinforcement learning, symbolic planning, and graph neural networks. Consistent with the perspective of interwoven production dimensions (cf. Hossfeld et al. (2026)), the platform does not impose a fixed hierarchy among objectives, but instead exposes time, energy consumption, reliability, and—by extension—risk as co-evolving quantities evaluated relative to the current system state.

3.2. *Factory Simulation Layer*

The factory simulation layer represents a virtual factory environment that simulates the transportation and processing of products by machines in discrete time steps. An object-oriented design is employed, where machines of different types are encapsulated within a factory object, and both factory and product objects are contained within a scenario object. Machines provide one or more skills, while products require one or more processing tasks for completion. Each task requires at least one compatible skill and may be executed by multiple alternative skills. A simplified high-level class diagram is shown in Fig. 2.

3.2.1. *Factory, Machines and Skills*

In this paper, a factory is defined as a set of machines, each offering one or more skills. As shown in Fig. 3, machines are classified into stationary machines and transporter machines. Stationary

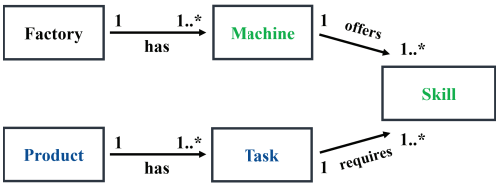


Fig. 2. Simplified class diagram of the associations between machines, skills, products, and tasks.

machines store, retrieve, or process products (e.g., drilling machines), while transporter machines move products between stationary machines, such as automated guided vehicles (AGVs) or conveyor belts. This structure allows the factory to be represented as a graph with stationary machines as nodes and edges defined by transport connections via transporter machines. These connections depend on the factory configuration and transporter type; for example, a conveyor belt may connect fewer machines than an AGV. Stationary machines are assigned fixed positions in a two-dimensional space, and for simplicity, straight-line distances between them are used as edge weights, although transportation distances may also be defined differently or specified manually.

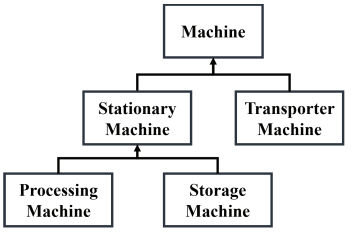


Fig. 3. Overview of the abstract machine types.

As stated before, each machine offers one or more skills. The skills are instances of the skill types shown in Fig. 4. Drilling would be an example of a processing skill. Transport and storage skills do not necessarily need to be further defined in this approach.

The goal of the proposed approach is the development of a modular platform for multi-objective optimization, considering the minimization of time and energy consumption and the maximization of reliability. To capture individual machine characteristics, each skill is associated with a time factor, an energy factor, and a reliability value. The time and energy factors determine the execution cost of processing and transport actions; for example, one machine may perform drilling quickly at high energy cost, while another operates more slowly but energy-efficiently. For transport actions, time and energy consumption may be computed by multiplying the corresponding skill factors with the distance between stationary machines. Reliability is defined as the probability of successfully processing or transporting a product and is modeled at the level of individual skills, following recent work on reliability assessment for software-defined and AI-controlled manufacturing systems Grimmeisen et al. (2022).

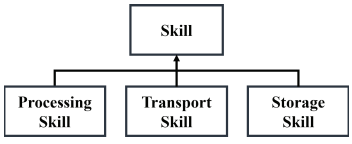


Fig. 4. Overview of the abstract skill types.

3.2.2. *Products and Tasks*

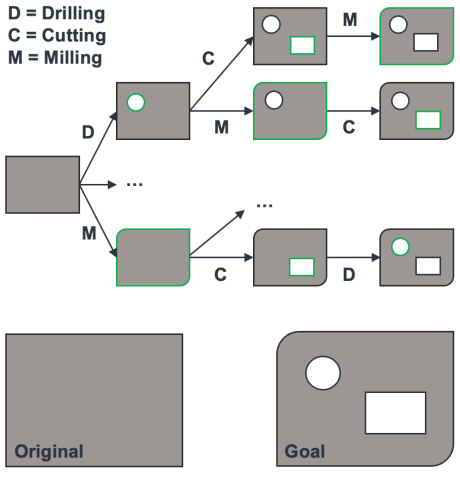


Fig. 5. An example of the sequential application of drilling, cutting, and milling skills to achieve the goal state of the product.

A task is an action that must be performed on a product within a factory in order to reach its predefined target state, which comprises both the set of completed processing tasks and the final product location. As illustrated in Fig. 5, a metal plate may require the sequential application of drilling, cutting, and milling skills to reach its goal configuration. A product is considered complete only once all required processing tasks have been successfully executed and the plate has reached its designated final state.

The abstract task types are shown in Fig. 6. Processing tasks are directly associated with a specific product, as they modify its physical properties or geometry, whereas transport tasks are generally available for all products. Consequently, transport tasks are associated with the factory itself, since transporter machines are defined at the factory level rather than being tied to individual products.

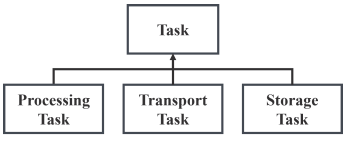


Fig. 6. Overview of the abstract task types.

Each task requires a skill and may be associated with multiple compatible skill types (cf. Section 3.2.1); for example, drilling may be executed using either a drilling or milling skill. Task instances include parameters used to compute execution time and energy, which are calculated using task-specific logic and the selected skill's time and energy factors. Task execution success is determined by the reliability of the selected skill via pseudo-random sampling. The result of a task execution is stored in a `TaskResult` object containing the product, the machine and skill used, the executed task, the incurred time and energy consumption, and a boolean success flag.

3.3. *Interface Layer*

The interface layer is located between the factory simulation layer and the scheduling method layer. It abstracts machines, skills, products, and tasks by providing information on the current simulation state and the set of available actions. As a result, scheduling methods are independent of the concrete factory and product definitions and operate solely on state representations and action sequences. This design facilitates switching between different scheduling methods and enables fair comparison, as all methods rely on the same state and action information.

3.3.1. *Action Catalog and Masking*

In the context of this paper, an *action* is defined as the application of a machine skill to a product in order to complete a task and thereby change the product's state. An *action key* is a tuple consisting of the unique identifiers of a product, a task, and a skill. The *action catalog* is defined as the sorted set of all possible action keys for a given factory and product configuration.

Algorithm 1 Construction of the Action Catalog

Initialize empty action catalog $\mathcal{A}$
for all products p **do**
  for all processing tasks t required by p **do**
    for all skills types s_t compatible with t **do**
      for all machines m providing skill s_t **do**
        Let s_m be the skill instance of m
        Add action key (p, t, s_m) to $\mathcal{A}$
      end for
    end for
  end for
for all transport tasks t in the factory **do**
  for all transport machines m **do**
    Let s_m be the transport skill of m
    Add action key (p, t, s_m) to $\mathcal{A}$
  end for
end for
Remove duplicate action keys from $\mathcal{A}$
Sort $\mathcal{A}$ by product, task, and skill identifiers
return $\mathcal{A}$

Action catalog: The construction of the action catalog is formalized in Algorithm 1. The algorithm iterates over all products in the factory and considers all required processing tasks of each product. For each processing task, all compatible skill types are identified, and for each skill type, all machines that provide such a skill are considered. The corresponding skill instance is

combined with the product and task identifiers to form an action key. In addition, transport actions are generated by combining each product with all transport tasks and transport skills available in the factory. After all action keys have been generated, duplicates are removed and the catalog is sorted to ensure a consistent ordering. This consistency is essential, as the action catalog must remain unchanged as long as the factory and product configurations remain the same.

Feasible actions: Not all actions contained in the action catalog are executable in every state of the system. For example, a product may only be processed by a machine if it is currently located at that machine. Such state-dependent constraints define the set of *feasible actions* for a given state. The feasible action set is computed in a similar manner to the action catalog, but restricted to the current state: only remaining processing tasks of each product are considered, and only transport tasks originating from the product's current location are included.

Action mask: An *action mask* is defined as a binary vector with the same length as the action catalog and is initialized with zeros. For each feasible action, the corresponding entry in the action mask is set to one. This mask explicitly encodes which actions from the global action catalog are admissible in the current state. The relationship between the action catalog and the action mask is illustrated in Fig. 7.

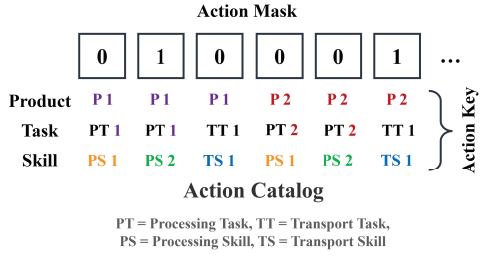


Fig. 7. Concept of an action catalog and action mask.

3.3.2. *State Encoding*

The state of a product is defined by its completed processing tasks and its current location, as described in Section 3.2.2. To abstract from object representations, a binary array is created for each product, initialized with zeros. The array contains one bit per processing task followed by one bit per machine in the factory. During state updates, bits corresponding to completed tasks and the current product location are set to one. The resulting two-dimensional array encodes the state of all products and can be flattened into a one-dimensional bit vector. This representation is computationally efficient and memory-light, which is beneficial for computationally intensive scheduling methods such as reinforcement learning that require explicit state information.

3.4. *Scheduling Method Layer*

The scheduling method layer comprises a set of interchangeable algorithms used to optimize action execution within the factory. The objective is to identify action sequences that complete all products while minimizing total execution time and energy consumption and maximizing reliability. The scheduling algorithms themselves are not contributions of this paper; their detailed formulation and evaluation are beyond its scope. Instead, the focus lies on their integration into a common framework and demonstrating their applicability within the proposed platform. Two representative scheduling approaches are outlined below.

3.4.1. *Metaheuristic Algorithm*

The Non-dominated Sorting Genetic Algorithm (NSGA-II) is used as an example of a metaheuristic scheduling approach. Each individual encodes a fixed-length sequence of actions, where an action corresponds to executing a processing or transport task using a machine skill. The interface layer provides a state-dependent feasible action catalog (cf. Subsection 3.3.1), enabling the generation of random but feasible action sequences for the initial population, which may not necessarily complete all products. NSGA-II is applied to identify action sequences that minimize execution time and energy consumption while maximizing reliability. Sequence reliability is computed as the product of individual action reliabilities, assuming independent execution, and is incorporated into

the minimization objective as one minus the sequence reliability.

3.4.2. *Reinforcement Learning*

As an AI-based scheduling approach, a Deep Q-Network (DQN) trained via reinforcement learning is employed. The agent is provided with the full action catalog, a state-dependent action mask (cf. Subsection 3.3.1), and the encoded state of all machines and products (cf. Subsection 3.3.2). After each action, the agent receives a reward equal to the negative sum of the action's execution time and energy consumption. Reliability is incorporated by subtracting a penalty proportional to one minus the reliability of the selected skill, scaled by a factor of 100 to ensure sufficient sensitivity among highly reliable skills. Additional rewards are granted for completing processing tasks, individual products, and all products, encouraging the agent to learn action sequences that balance time, energy, and reliability while completing all products.

3.5. *Scenario Definition*

Scenarios are defined using JSON files that specify both the factory and the products to be manufactured. The scenario description includes all machines in the factory (processing, storage, and transport), their available skills, and the required processing tasks for each product, along with their initial and target locations. Each skill is parameterized by individual time and energy factors as well as a reliability value, enabling scenario-specific trade-offs between performance and dependability. A small excerpt of the scenario definition is shown below:

```
"processing_machines":[
 {"class":"MillingMachine","name":"MM2",
  "skill_params":{"milling_skill":{
   "time_factor":1.2,"energy_factor":0.9,
   "reliability":0.995}}}
],
"product":{"class":"Plate",
"starting_location":"ST1",
 "target_location":"ST1",
 "processing_tasks":[
  {"class":"DrillingTask"},
  {"class":"MillingTask"},
  {"class":"CuttingTask"}]}
```

4. Demonstrative example

4.1. *Factory Configuration*

The factory comprises three types of processing machines, each providing a single skill for simplicity: drilling machines (DM) with a drilling skill (DS), milling machines (MM) with a milling skill (MS), and cutting machines (CM) with a cutting skill (CS). In addition, one automated guided vehicle (AGV) serves as a transport machine (TM) and provides a transport skill (TS), enabling transportation between any two stationary machines and resulting in 42 available transport tasks. A generic storage machine (SM) providing a storage skill (SK) is also included and serves as both the initial and target location for all products.

The case study uses two instances of each processing machine type, one transport machine, and one storage machine. For demonstration purposes, all skills have time and energy factors set to 1.0, ensuring that scheduling outcomes are not influenced by these parameters. All machines are assumed to be fully reliable (skill reliability 1.0), except for one milling machine: the milling skill of MM2 (MS2) is assigned a reduced reliability of 0.995 to evaluate the impact of reliability-aware scheduling. The layout of stationary machines and their connections via the AGV is shown in Fig. 8.

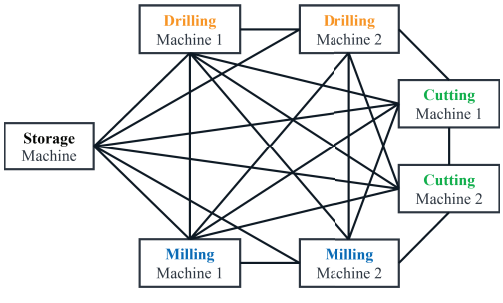


Fig. 8. Top-down view of the stationary machines within the factory and the connections between them.

4.2. *Product Configuration*

The product considered in this case study is a metal plate that undergoes drilling, milling, and cutting operations, as illustrated in Fig. 5. The

plate starts at the storage machine SM1, is processed by the available machines, and must be returned to the same storage machine once all processing tasks are completed. The required processing tasks comprise one drilling task (DT1), one milling task (MT1), and one cutting task (CT1). The parameters used to compute absolute time and energy consumption are omitted, as they are not relevant for the comparative evaluation performed in this paper. Instead, task-specific values are scaled by the time and energy factors of the selected skills, as described in Subsection 3.2.2. Since these parameters are independent of the chosen skill, time and energy consumption are directly proportional to the corresponding skill factors. While omitted here, task parameters may be included in future studies to enable more fine-grained analyses, for example when considering multiple tasks of the same type with different characteristics.

5. Results and Discussion

The modular platform presented in this paper was used to compare a heuristic scheduling approach based on NSGA-II with an AI-based scheduling approach using a DQN trained via reinforcement learning. Both scheduling methods operated exclusively on the abstractions provided by the interface layer and were fully decoupled from the factory simulation layer. As a result, both approaches optimized identical state representations and action spaces, ensuring a fair and reproducible comparison. To mitigate stochastic variability, ten independent runs were conducted on the same standard consumer-grade desktop computer. The average results are presented in Table 1.

Table 1. Average results from ten independent runs.

Scheduling Approach	Actions (Average)	Cost (Average)	Duration (Average)
NSGA-II	7.0	19.09	2min 2s
DQN	7.0	19.14	5min 21s

Note: Duration of DQN training, not inference. Cost function based on execution time, energy consumption, and reliability.

Both scheduling methods were able to identify valid sequences of actions that completed the product. In the considered scenario, the resulting action sequences were minimal in length and appeared minimal with respect to the cumulative execution time and energy consumption until product completion. This indicates that the platform enables heterogeneous optimization methods to converge toward comparable solutions under identical modeling assumptions. In terms of computational effort, the NSGA-II-based scheduling approach was faster. However, these results are not intended as a performance benchmark between optimization methods and are highly dependent on the complexity of the scenario.

With respect to reliability, both the NSGA-II–based and the reinforcement learning–based approaches consistently avoided the milling skill with reduced reliability in favor of the more reliable alternative over multiple independent runs. This behavior demonstrates that reliability attributes assigned at the skill level are effectively propagated through the planning process and influence scheduling decisions in both classical and learning-based methods. The results confirm that the platform supports reliability-aware planning without requiring method-specific adaptations.

From a conceptual perspective, these results illustrate how production behavior emerges from state-dependent orchestration of skills rather than from a predefined process structure. The consistent avoidance of less reliable skills by both optimization approaches demonstrates how reliability attributes directly influence the realized production configuration, supporting the view of production systems as dynamically instantiated rather than statically defined.

Although the demonstrative example employs a simplified reliability-based representation of risk, it highlights how risk-related attributes can be exposed uniformly across heterogeneous scheduling approaches and can shape production outcomes through planning decisions. This illustrates the suitability of the platform as a controlled experimental environment for studying trade-offs between efficiency, reliability, and risk under consistent conditions.

6. Conclusion

This paper presented a modular platform for multi-objective scheduling in skill-based manufacturing systems, with an explicit focus on integrating time, energy consumption, and reliability considerations. By decoupling factory simulation from scheduling algorithms through a common interface, the platform enables different optimization methods to be applied and compared.

It was demonstrated that both a meta-heuristic approach based on NSGA-II and an AI-based approach using reinforcement learning can generate valid production sequences and account for reliability differences between machines. The results indicate that the proposed platform supports meaningful comparison of heterogeneous scheduling strategies while ensuring consistency in state representation and action availability.

Future work will focus on extending the range of evaluated scheduling methods, increasing scenario complexity, and conducting more systematic performance and scalability analyses. In addition, the integration of more detailed risk models and failure mechanisms will be investigated to further strengthen the platform's applicability to safety-critical manufacturing systems. The developed framework is open-source and publicly available at `https://github.com/mbsa-tud/ARISE`. In this sense, the proposed platform can be understood as an operational step toward the concept of formless production (Hossfeld et al. (2026)), providing a concrete and reproducible means to explore how time, energy consumption, reliability, and risk jointly shape emergent production behavior under dynamic conditions.

Acknowledgement

This research has been funded by the Ministry of Science, Research and Arts of the Federal State of Baden-Württemberg within the InnovationsCampus Future Mobility, `www.icm-bw.de`.

References

Aupy, G., A. Benoit, and Y. Robert (2012). Energy-aware scheduling under reliability and makespan constraints. In *2012 19th International Conference on High Performance Computing*, pp. 1–10.

Burmeister, S. C., T. N. Rogalski, and G. Schryen (2025). Comparative analysis of evolutionary algorithms for energy-aware production scheduling. *arXiv preprint arXiv:2504.15672*.

Froschauer, R., A. Köcher, K. Meixner, S. Schmitt, and F. Spitzer (2022). Capabilities and skills in manufacturing: A survey over the last decade of etfa. In *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*, pp. 1–8. IEEE.

Grimmeisen, P., R. Golwalkar, F. Sautter, and A. Morozov (2025). Relaibotix: Reliability assessment for ai-controlled robotic systems. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5496–5503.

Grimmeisen, P., A. Morozov, T. Fabarisov, A. Wortmann, and C. H. Koo (2022). Automated model-based reliability assessment of software-defined manufacturing. In *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*, pp. 1–4.

Hossfeld, M., A. Morozov, and A. Wortmann (2026). Towards formless production with skill-based industrial digital twins. In *Proceedings of the IEEE CCNC 2026 Conference*.

Hossfeld, M. and A. Wortmann (2024). A universal framework for skill-based cyber-physical production systems. *Journal of Manufacturing and Materials Processing 8*(5), 221.

Meilanitasari, P. and S.-J. Shin (2021). A review of prediction and optimization for sequence-driven scheduling in job shop flexible manufacturing systems. *Processes 9*(8).

Módos, I., P. Šůcha, and Z. Hanzálek (2017). Algorithms for robust production scheduling with energy consumption limits. *Computers & Industrial Engineering 112*, 391–408.

Pfrommer, J., D. Stogl, K. Aleksandrov, S. E. Navarro, B. Hein, and J. Beyerer (2015). Plug & produce by modelling skills and service-oriented orchestration of reconfigurable manufacturing systems. *at-Automatisierungstechnik 63*(10), 790–800.

Quan, Z., Y. Wang, and Z. Ji (2022). Multi-objective optimization scheduling for manufacturing process based on virtual workflow models. *Applied Soft Computing 122*, 108786.

Rastgar, I., J. Rezaeian, I. Mahdavi, and P. Fattahi (2023, 11). Modeling and solving the multi-objective energy-efficient production planning and scheduling with imperfect maintenance activities. *Journal of Quality in Maintenance Engineering 30*(1), 26–50.

Zhang, W., Y. Zheng, and R. Ahmad (2023). An energy-efficient multi-objective integrated process planning and scheduling for a flexible job-shop-type remanufacturing system. *Advanced Engineering Informatics 56*, 102010.