Reactive Disassembly Sequence Planning under Uncertain Component Conditions

Wael El Morabit * **Mohammed-Amine Abdous** *
Flavien Lucas * **German Bluvstein** **
Finn-Augustin Brunnenkant ** **Lasse Streibel** **

* *IMT Nord Europe, Institut Mines-Télécom, Univ. Lille, Center for Digital Systems, F-59000 Lille, France (e-mail: {wael.elmorabit, amine.abdous, flavien.lucas}@imt-nord-europe.fr).*
** *Technical University of Munich, TUM School of Engineering and Design, Institute for Machine Tools and Industrial Management (iwb), 85748 Garching, Germany (e-mail: {german.bluvstein, finn.brunnenkant, lasse.streibel}@iwb.tum.de).*

Abstract:
Although disassembly supports circular production, planning remains difficult because the true condition of components is often unknown, and precedence relations create a large combinatorial search space. We address this problem in the setting of reactive disassembly sequence planning. Products are modeled as directed acyclic graphs with processing time, cost, and recovery profit at the component level for both complete and selective disassembly. A mixed-integer linear program provides a reference model on moderate-size instances, and a hybrid planner that combines Greedy Randomized Adaptive Search Procedure (GRASP) with Variable Neighborhood Descent generates a feasible sequence at scale. On 246 benchmark instances with 10 to 1,000 components, the hybrid planner stays within 1% of the mixed-integer reference on average, with solution times below 0.5 s per instance. To cope with deviations during execution, we add a fuzzy layer that monitors simple process signals such as force or torque spikes, repeated vision failures, timeouts, and loss of grip. It selects among four actions. It can either bypass the component, perform a tool change and retry, or carry out controlled destruction or local replanning on the residual graph, updating the sequence accordingly. Failure simulations on 90 scenarios show that the adaptive pipeline maintains a feasible plan and recovers all targets in most runs, and partial recovery in the remaining cases, with response times compatible with real-time use. For end-of-life disassembly cells, this provides a practical support tool that turns optimized plans into simple shop-floor rules on how to react when a step fails, helping maintain flow and recovery profit without constant expert intervention.

Keywords: Disassembly sequence planning, Reactive disassembly planning, Hybrid metaheuristics, Circular production and remanufacturing, Sustainable manufacturing.

1. INTRODUCTION

Circular production depends on the ability to recover parts and materials at the end of life. Disassembly makes this possible by disassembling components that can be reused, repaired, or recycled (Sundin et al., 2012). In practice, disassembly is challenging as the true state of parts is often unknown at the start of the disassembly process. Execution can reveal deviations such as hidden wear, seized or missing fasteners, misalignment, and minor damage. These factors increase time and cost and create stops on the line.

To remain efficient, disassembly systems must adapt disassembly plans when deviations are observed. This approach is known as reactive disassembly planning and may involve process or schedule adaptations. Due to the complex, uncertain, and dynamic nature of the decision-making environment of reactive disassembly planning, data-driven approaches are necessary (Streibel et al., 2024).

In this context, disassembly sequence planning (DSP), which is a subdiscipline of disassembly process planning, is especially challenging due to its large combinatorial complexity. DSP refers to the search for all possible disassembly sequences and the selection of the optimal solution from them (Lambert, 2003).

Figure 1 places this contribution inside a broader architecture. Block one is sequence planning, which proposes feasible orders for a new product. Block two is reactive process planning, which adapts the disassembly sequence and local operations when a complication arises. These two blocks are implemented and analysed in this paper. Block three is AI-based automation that uses sensor data to handle uncertainty in stations (Bauer et al., 2023; Abdous et al., 2025). Block four is an AI-based inspection that assesses the quality of recovered parts. All blocks share knowledge through a common layer for information exchange and learning, and the methods developed here are intended as a first step toward this integrated architecture. As disassembly information is often stored in different formats and can be unstructured, the proposed knowledge base should follow a clearly defined schema that structures the information and connects it across all sources. Such a unified schema makes it possible to integrate data from different file formats (e.g., CAD models and manuals) into a single knowledge base, where all required knowledge for disassembly planning is represented in a consistent way (Bluvstein and Daub, 2026).

This work makes four contributions. First, a simple and general model that links cost, profit, and time at the component level for both complete and selective disassembly. Second, a scalable planner that delivers near-optimal value with very short run time on large products. Third, a compact fuzzy decision layer with four clear actions that planners can tune and explain. Fourth, managerial insight that shows how the method can increase net recovery profit, lower operating cost through less rework and downtime, and improve plan reliability.

The remainder of this paper is organized as follows. Section 2 reviews related work on disassembly planning, reactive execution, and fuzzy decision rules. Section 3 introduces the problem setting and the reactive execution scheme. Section 4 presents the benchmark design, failure scenarios, and numerical results. Section 5 summarises the main findings and outlines directions for future work.

2. LITERATURE REVIEW

Disassembly Sequence Planning (DSP) concerns the computation of an order of operations to dismantle end-of-life products subject to structural and precedence constraints, with objectives that typically combine value recovery, cost, and time. DSP has progressed from deterministic formulations to uncertainty-aware and adaptive decision-making frameworks. Several recent surveys and application-driven studies consolidate these trends and articulate open challenges in modeling, scalability, and uncertainty handling. Guo et al. (2021) reviews DSP models, objectives, and algorithms, highlighting the shift toward scalable hybrid solvers and uncertainty-aware formulations. Streibel et al. (2025) provides a comprehensive review of data modeling methods for multiple research fields in disassembly planning and control, including DSP and reactive disassembly planning. For DSP, products are commonly represented by graph-based models (e.g., directed acyclic graphs, AND/OR graphs) that encode feasible removal relations and optionality (Streibel et al., 2025). The resulting combinatorial search is NP-hard, with exact models for optimality proofs on moderate sizes and scalable approximation via heuristics and metaheuristics.

Mixed-Integer Linear Programming (MILP) is still used as an exact baseline on small and medium instances but does not scale well because of combinatorial growth and stochastic effects (Bentaha et al., 2015a). For larger problems, most studies turn to hybrid metaheuristics that combine a constructive phase with local improvement, often GRASP with Variable Neighborhood Descent and related neighborhoods to refine precedence-feasible sequences (Zhou et al., 2019). Population-based methods (GA, PSO, ACO) are also reported, but single-solution hybrids are often preferred because they are simpler to implement, robust, and easier to adapt to problem-specific feasibility checks.

For the consideration of uncertainty and random operation times, Chance-constrained formulations ensure probabilistic feasibility when processing times or success probabilities are stochastic (Tian et al., 2013). Fuzzy models capture linguistic or poorly identified quality information and varying operational costs, enabling robust sequence choices without strong distributional assumptions (Tian et al., 2018). Selective DSP integrates these ideas by targeting only high-value or high-quality components when full disassembly is unnecessary, often under random durations and yields (Kim et al., 2018). Together, these works establish stochastic and fuzzy programming as complementary tools for realism and robustness.

A substantial stream of work addresses disassembly line balancing and sequencing when task times are uncertain. Building on precedence models such as AND/OR and DAG structures, Bentaha et al. (2013) proposed a stochastic formulation that explicitly embeds randomness at the modeling stage. Bentaha et al. (2015a) developed an exact approach for line balancing with stochastic processing times, providing optimality guarantees on moderate-size instances and a baseline for heuristic and metaheuristic methods. For a broader overview, Bentaha et al. (2015b) reviewed modeling choices, stochastic and robust formulations, solution paradigms, and highlighted uncertainty-aware line-level coordination as a key requirement for scalable disassembly systems.

In this paper, we adopt the same general architecture. Products are modeled as directed acyclic graphs with economic data at the component level. A hybrid planner based on GRASP with Variable Neighborhood Descent then produces scalable sequences for both complete and selective disassembly. On top of this nominal planner, a fuzzy reactive layer monitors simple process signals and applies local actions such as bypass, tool change, controlled destruction, and local replanning. Together, these elements form a reactive disassembly process planning framework that connects offline optimization with online adaptation under uncertain component conditions.

3. PROBLEM SETTING AND REACTIVE EXECUTION SCHEME

We study disassembly sequence planning for a single product in an industrial context where component conditions are uncertain. The product is modeled as a directed acyclic graph $G = (V, E)$. Each node $i \in V$ represents a disassembly step. Each arc $(i, j) \in E$ indicates that component i must be removed before component j becomes accessible.

For every node i we associate a processing time t_i, an operational cost c_i, and a recovery profit p_i. These quantities define the economic contribution of the component. A target set $T \subseteq V$ identifies the components that must be recovered at the end of the process. When $T = V$, the plan corresponds to complete disassembly. When T is a strict subset of V, the plan corresponds to selective disassembly, and noncritical components may be left in place if this does not block any target.

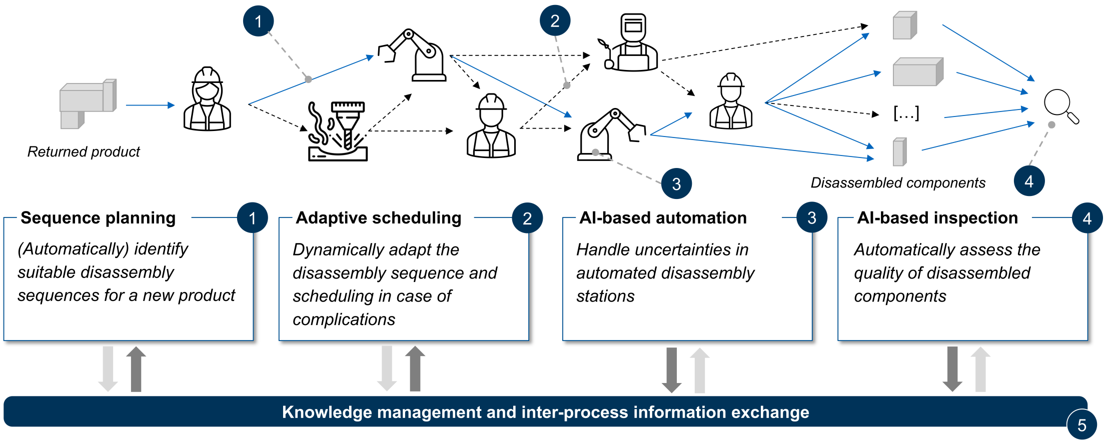

Fig. 1. End-to-end architecture for reactive disassembly. Block 1 performs sequence planning for new products, Block 2 carries out reactive process planning when complications occur, Block 3 uses AI-based automation to handle uncertainties in disassembly stations, and Block 4 applies AI-based inspection to recovered components. Block 5 provides a shared knowledge layer that links all blocks through information exchange.

A disassembly sequence is an ordering of nodes that respects all precedence relations in G and satisfies the mode defined by T. Executing such a sequence under nominal assumptions leads to the removal of all targets with the associated cost and time. In practice, the true state of each component is not fully known at planning time. During execution, the process may reveal hidden wear, seized or missing fasteners, misalignment, or minor damage. Sensing issues such as repeated vision failure, abnormal force or torque values, and loss of grip also occur. These events can delay or block some operations and may invalidate the nominal sequence.

The objective of the planner is to construct a precedence feasible sequence that maximizes the net recovery profit. This profit is computed from recovered components, processing and destruction costs, and penalties for excessive time or rework. At the same time, the planner must provide a mechanism to adapt the sequence when unexpected events appear. The approach developed in this paper, therefore, combines an offline planning stage with an online adaptation stage, both based on the same graph and cost-profit model.

3.1 Adaptive execution scheme

Figure 2 presents the execution scheme used in the proposed framework. It links the offline planner and the online decisions taken during disassembly.

The upper part of the diagram is the nominal path. The input is the product graph with its target set of components. The offline planner computes a reference sequence $\sigma^\star$ with a GRASP constructor and local improvement by Variable Neighborhood Descent. During execution, the controller follows this sequence. After each operation, it updates the product state and checks whether all required components have been processed. If this is the case, the process stops, and performance indicators are computed. If not, the controller moves to the next component in $\sigma^\star$. At every step, the controller also monitors simple indicators of component condition and station behaviour, such as abnormal torque, repeated vision failures, timeouts, and loss of grip. When no symptom is detected, the system stays on the nominal path and executes the next component in $\sigma^\star$.

When a symptom is detected, the flow switches to the adaptive path in Figure 2. A failure detection block aggregates raw sensor signals into a compact symptom description. This description, together with information on the current component and on the remaining plan, is sent to a Mamdani-type fuzzy inference system (Mamdani, 1977). The fuzzy rules produce preferences for four actions. The controller can bypass a noncritical component, change the tool or method, and retry, apply a controlled destructive removal when the expected loss in value is acceptable, or call a local replanning routine. The selected action is then dispatched, and the graph and sequence are updated if needed.

If the action does not solve the problem, the fallback path is activated. The remaining part of the graph and the current state of execution define a reduced instance of the planning problem. The planner is applied to this reduced instance to compute a new sequence. This sequence replaces the unused part of $\sigma^\star$ and the controller resumes execution on the nominal path. The three paths in Figure 2 thus form a closed loop scheme that follows a planned sequence when possible and adapts it locally when component conditions deviate from the nominal assumptions.

From a control point of view, the fallback path enforces two simple constraints. It never revisits operations that are already completed, so all replanned sequences remain consistent with the physical state of the product. It also keeps the optimization scope local by replanning only the remaining part of the graph. This limits computation time and reduces the number of changes that operators see during execution. Because the same objective and constraints are used in the nominal planner and in the fallback planner, the economic logic of the initial sequence is preserved even after several fallback steps.

At the decision level, the fuzzy layer uses four inputs: the estimated failure probability p_{fail}, the measured force and torque, and a context score that reflects the availability of

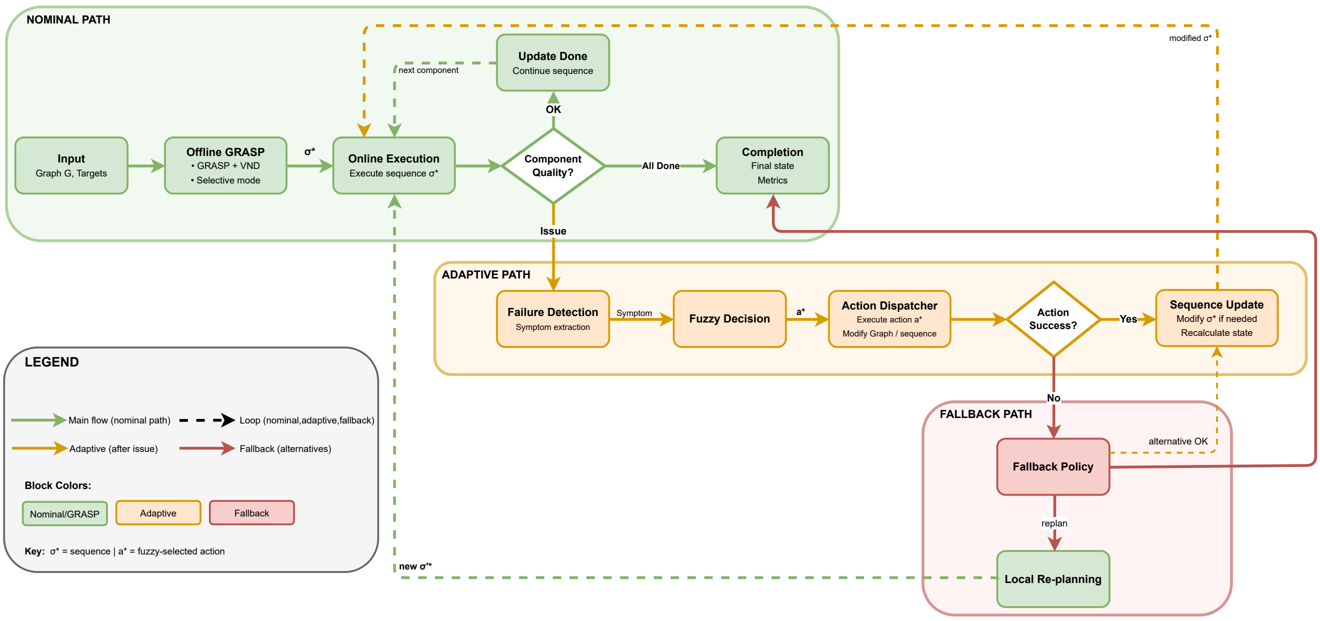

Fig. 2. Adaptive execution architecture for disassembly sequence planning. The nominal path executes the GRASP sequence, the adaptive path applies fuzzy actions after detecting symptoms, and the fallback path performs local replanning on the remaining graph.

alternative tools or paths. The rule base comprises nine Mamdani rules that cover typical failure patterns (Mamdani, 1977). When the context score indicates that alternatives are available, or when p_{fail} is low with normal force and torque, the system prefers a tool change and retry. Medium failure probability with low or medium loads leads to bypass decisions, while high loads with low failure probability favour bypassing a single difficult component. High failure probability alone triggers local replanning on the residual graph, and high failure probability combined with high loads activates controlled destruction when the economic conditions are satisfied. These rules are applied through a standard fuzzy pipeline with fuzzification of the inputs, rule activation, aggregation of the outputs, defuzzification, and selection of a single action at each decision step.

4. EXPERIMENTATION

4.1 Benchmark performance of the nominal path

The nominal path is a hybrid metaheuristic that follows a standard GRASP with local search. Each iteration starts from an empty sequence and builds a complete precedence feasible sequence. At every construction step, the candidate set contains all currently accessible components. A greedy function that combines marginal profit and processing time ranks these candidates, and a restricted candidate list is formed. One candidate is drawn at random from this list and appended to the partial sequence. After construction, a Variable Neighborhood Descent phase applies swap, insert, and reverse moves while feasibility is preserved, until no improving move is found. The best solution over all iterations is returned.

The nominal path was first evaluated on a benchmark of 246 generated instances. The number of components ranges from 10 to 1,000 and both complete and selective disassembly modes are considered. For each instance, we run the hybrid planner. For the subset of instances that the MILP reference can solve without exhausting memory, we also record the net recovery profit and the computational time returned by Cplex (Lambert and Gupta, 2008). On this solvable subset of the benchmark, the average profit gap between the hybrid planner and the MILP is less than 1%, and the average time needed to obtain a solution is below 0.5 seconds per instance for both complete and selective disassembly modes.

4.2 Performance under failure scenarios

To assess the robustness of the adaptive layer, we ran failure simulations on 30 instances taken from the full benchmark, with graph sizes between 7 and 148 nodes. For each instance, 3 failure scenarios of increasing difficulty were defined, leading to 90 runs in total. In simple scenarios, the failure is placed on a non-critical node, so that at least one alternative path to the main targets remains. In intermediate scenarios, the failure affects a component of medium importance. The target remains reachable, but the system must modify the planned sequence, for example, by taking a less profitable path or by adding extra actions before it can reach the targets. Complex scenarios place the failure on a critical node that lies on every path to at least one target, so that the target may become unreachable unless destructive removal or local replanning on the residual graph can restore feasibility.

Across the 90 runs, the adaptive pipeline always maintains a consistent graph and sequence. All 30 simple scenarios and 29 of 30 intermediate scenarios reach all targets, giving success rates of 100% and 97% for these difficulty levels. In complex scenarios, 27 of 30 runs still recover the full target set, and in the remaining 3 runs, failures located on

Table 1. Adaptive actions on the example instance

Node	Force	Torque	Symptom and action
N07	60.0	70.0	Over-torque, destruction
N06	30.0	35.0	Over-torque, change tool
N17	48.0	52.0	Access blocked, bypass
N21	75.0	80.0	Low value, destruction

topologically critical components lead to a partial recovery of the targets. Overall, 86 of 90 runs achieve 100% target recovery, while the 4 remaining runs (1 intermediate and 3 complex) retain between 33% and 67% of the requested targets instead of failing.

The adaptive layer reacts with a small number of local actions. Simple and intermediate scenarios typically require between 1 and 4 adaptive actions, while the most stressed complex cases use up to 6 actions, including bypass and local replanning. End-to-end execution time, including failure detection, action selection, and possible replanning, ranges from 0.02 s to 0.65 s and stays below 0.2 s on most instances. This indicates that the adaptive behaviour is compatible with near real-time use at the cell level.

4.3 Illustrative instance

We illustrate the behaviour of the adaptive layer on a selective disassembly instance with twenty-eight nodes. Figure 3 shows the product graph and the status of each node at the end of the run. Green nodes belong to the nominal sequence, orange nodes correspond to operations where an adaptive action was applied, the red-framed node indicates a fallback with replanning, and the blue node is the target component.

The nominal sequence produced by the planner starts with nodes $N01$, $N03$, $N05$ and $N07$ and so on till ending with target $N22$. While performing the disassembly steps, the execution follows this sequence until node $N07$, where an over-torque symptom is detected. The fuzzy layer suggests destruction, but this is not allowed because the component has a high recovery value. The fallback path is therefore activated, no other alternative is found, so we destroy the $N07$, and lose its value. A new over-torque symptom appears on node $N06$, in this case, the fuzzy layer selects the change tool action, and the operation succeeds. At node $N17$, access to the component is blocked, and the fuzzy layer proposes a bypass since it appears to be possible, the controller bypasses this node. At node $N21$, a low-value component creates a bottleneck. The fuzzy layer proposes controlled destruction, and the economic threshold is satisfied, so destruction is applied. After a short replanning step, the sequence ends with the removal of the target node $N22$.

Table 1 summarises the main adaptive events. For each affected node, we report the measured force and torque, the symptom detected, the action selected by the fuzzy layer, and the observed result.

In this run, some symptoms are handled by simple local actions such as changing the tool or bypassing a noncritical component, while others require a fallback and replanning on the residual graph. The target component is reached in all cases, and the plan remains feasible.

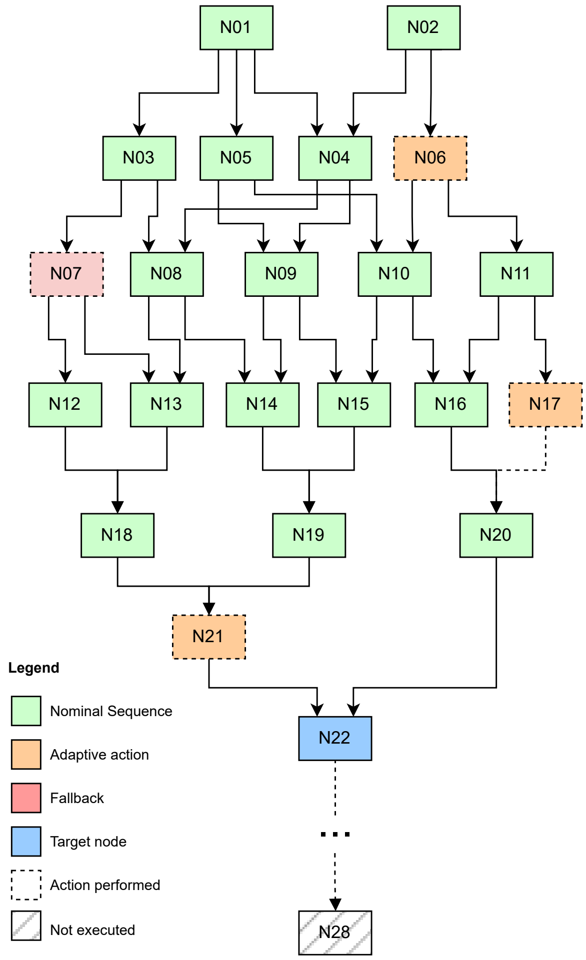

Fig. 3. Example instance: Green nodes form the nominal sequence. Orange nodes correspond to adaptive actions. The red-framed node indicates a fallback with replanning and the blue node is the target.

5. CONCLUSION AND PERSPECTIVES

This paper has introduced an adaptive framework for disassembly sequence planning under uncertain component conditions. The product is represented as a directed acyclic graph with cost, time, and profit attached to each component. A planner that combines a greedy randomized constructor with local improvement by Variable Neighborhood Descent produces precedence feasible sequences for both complete and selective disassembly. On a benchmark with up to one thousand components, the planner remains within a few percent of a reference method while keeping computation times low. Near-optimal economic performance can be obtained at a computational cost compatible with industrial use and with interactive planning.

The framework extends offline planning with an adaptive execution layer. A Mamdani type fuzzy decision system monitors simple symptoms and uses component criticality and remaining profit margin to propose local actions. Bypass, tool change, controlled destruction, and local replanning are applied when needed, and the fallback path reuses the same planner on reduced instances. The illustrative instance shows that this combination maintains feasibility and keeps the final profit close to the nominal value without manual intervention, even when several symptoms and changes of strategy occur along the sequence.

Several perspectives arise from the limitations of the present study. The product model is restricted to a directed acyclic graph with a single type of precedence relation. In practice, AND/OR structures often appear when optional parts and alternative disassembly paths must be represented. Testing the proposed framework on AND-OR graphs is a natural extension, but the potential gain in modeling power must be balanced against the added complexity in sequence generation and fuzzy rule design. Furthermore, the role of a clearly defined knowledge schema (e.g., ontologies and taxonomies) requires further investigation. While AND-OR graphs can capture the product structure, additional information is necessary to determine optimal disassembly sequences. Because this information is typically distributed across heterogeneous formats, a consistent schema is needed that structures and links all relevant data. Establishing such a schema will provide the basis for realizing the full pipeline outlined in Figure 1, from computer-aided design data to automatic graph construction, economic parameterization, planning, adaptive execution, AI-based station control, and AI-based inspection of recovered parts. In this view, the present work provides a core module that links planning and adaptation, and that can be connected to richer perception and decision support components in future circular production systems.

ACKNOWLEDGEMENTS

This work was carried out as part of the LAsSy project (Learning Assistance Systems for Efficient Disassembly), funded by the German French Academy for the Industry of the Future, whose support the authors gratefully acknowledge.

REFERENCES

Abdous, M.A., Delorme, X., Battini, D., and Sgarbossa, F. (2025). Scenario-based optimization and simulation framework for human-centered assembly line balancing. *International Journal of Production Economics*, 282, 109513.

Bauer, J.C., Abdous, M.A., Kurscheid, S., Lucas, F., Lozenguez, G., and Daub, R. (2023). Towards enabling human-robot collaboration in industry: Identification of current implementation barriers. In *5th Conference on Production Systems and Logistics*.

Bentaha, M.L., Battaïa, O., and Dolgui, A. (2013). A stochastic formulation of the disassembly line balancing problem. In *Advances in Production Management Systems. Competitive Manufacturing for Innovative Products and Services (APMS 2012)*, volume 397 of *IFIP Advances in Information and Communication Technology*, 397–404. Springer, Berlin, Heidelberg.

Bentaha, M.L., Battaïa, O., and Dolgui, A. (2015a). An exact solution approach for disassembly line balancing problem under uncertainty of the task processing times. *International Journal of Production Research*, 53(6), 1807–1818.

Bentaha, M.L., Dolgui, A., and Battaïa, O. (2015b). A bibliographic review of production line design and balancing under uncertainty. *IFAC-PapersOnLine*, 48(3), 70–75.

Bluvstein, G. and Daub, R. (2026). Information extraction and knowledge modeling for disassembly processes. In *Procedia Computer Science (7th International Conference on Industry of the Future and Smart Manufacturing (ISM 2025))*. In press.

Guo, X., Zhou, M., Abusorrah, A., Alsokhiry, F., and Sedraoui, K. (2021). Disassembly sequence planning: A survey. *IEEE/CAA Journal of Automatica Sinica*, 8(7), 1308–1324.

Kim, H.W., Park, C., and Lee, D.H. (2018). Selective disassembly sequencing with random operation times in a parallel disassembly environment. *International Journal of Production Research*, 56(24), 7243–7257.

Lambert, A.J.D. (2003). Disassembly sequencing: A survey. *International Journal of Production Research*, 41(16), 3721–3759.

Lambert, A.J.D. and Gupta, S.M. (2008). Methods for optimum and near optimum disassembly sequencing. *International Journal of Production Research*, 46(11), 2845–2865.

Mamdani, E. (1977). Application of fuzzy logic to approximate reasoning using linguistic synthesis. *IEEE Transactions on Computers*, C-26(12), 1182–1191.

Streibel, L., Brunnenkant, F.A., Reinbold, N., Lindholm, N., Jordan, P., Bluvstein, G., Zaeh, M.F., and Reuter, C. (2025). Systematic review of data modelling methods for system-level disassembly planning and control. *International Journal of Sustainable Engineering*, 18(1), 2563271.

Streibel, L., Jordan, P., and Zaeh, M.F. (2024). Data-driven approach for decision-making in reactive disassembly planning to enable case-based reasoning. *Procedia CIRP*, 130, 1117–1123. 57th CIRP Conference on Manufacturing Systems 2024 (CMS 2024).

Sundin, E., Elo, K., and Mien Lee, H. (2012). Design for automatic end–of–life processes. *Assembly Automation*, 32(4), 389–398. doi:10.1108/01445151211262447.

Tian, G., Zhou, M., and Chu, J. (2013). A chance constrained programming approach to determine the optimal disassembly sequence. *IEEE Transactions on Automation Science and Engineering*, 10(4), 1004–1013.

Tian, G., Zhou, M., and Li, P. (2018). Disassembly sequence planning considering fuzzy component quality and varying operational cost. *IEEE Transactions on Automation Science and Engineering*, 15(2), 748–760.

Zhou, Z., Li, J., Liu, Q., and Tian, G. (2019). Disassembly sequence planning: Recent developments and future trends. *Proc. IMechE Part B: Journal of Engineering Manufacture*, 233(5), 1450–1471.