



Final degree project

For obtaining the Master's Diploma

Field: Industrial Engineering
Major: Industrial Management and Logistics

Presented by :
TRARI MEDJAOUI Sara

Subject

**Metaheuristic Optimization
of a Multi-Type Human-Robot Collaborative
Disassembly Line
for End-of-Life Electric Vehicle Batteries**

Publicly defended, on 21/06/2026, in front of the jury composed of :

Mr. BRAHAMI Mustapha Anwar	MCA	ESSA. Tlemcen	President
Mr. MALIKI Fouad	MCA	ESSA. Tlemcen	Thesis Director
Mr. ARBAOUI Taha	Prof	INSA. Lyon	Co-Thesis Director
Mr. BENNEKROUF Mohammed	MCA	ESSA. Tlemcen	Examiner 1
Mr. HADDOU BENDERBAL Hichem	MC	Aix-marseille University	Examiner 2

Academic Year : 2025 / 2026

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

Dedication

I dedicate this work, first and foremost, to myself, for all the times I kept going despite fatigue, doubts, and difficult moments. To the version of myself that learned, grew, resisted, and continued until the end of this journey.

To my beloved parents, my father and my mother, to whom I owe everything. No words can truly express my gratitude for your love, sacrifices, patience, and endless support. You have always been my strength, my refuge, and my greatest source of motivation. If I have reached this point today, it is thanks to you, your prayers, your trust, and everything you have done for me.

To my dear sisters, Meryem and Yasmine, for your presence, love, support, and all the moments we have shared. Thank you for being by my side, each in your own way, through both good and difficult days.

To my grandparents, for their tenderness, prayers, and precious affection. To my entire family, for their love, encouragement, and constant presence throughout my journey.

To my friends Ghizlene, Hind, Rym, Kawther, and Nedra, with whom I shared a great part of my university journey. Thank you for the memories, the laughter, the support, the stressful moments, the achievements, and all the experiences that made these years more beautiful and lighter.

To my classmates in Industrial Engineering, with whom I shared the good and difficult moments, the long study days, the projects, the challenges, and the successes. These years will remain engraved in my memory because of you.

To the beautiful people I met throughout these years, who became an important part of my story. Thank you for all the moments of joy, sharing, and lightness: the lunch breaks, the spontaneous conversations, the laughter in the library, the fun moments, and all those little pauses that made this journey softer and more memorable. Even though we did not always share the same class or the same path, your presence meant a lot during these university years, and I am deeply grateful for it.

To all the members of the ECC club, thanks to whom I experienced some of the most beautiful years of my university life. This club gave me much more than activities: it allowed me to grow, learn, flourish, and above all, meet exceptional people who became true friends.

Finally, I dedicate this work to all my teachers, who contributed to my education and personal growth. I would especially like to mention Mr F.Maliki, a true pillar of our journey, for his dedication, guidance, and everything he has passed on to us. If I am here today, it is also thanks to all these people who have marked my path.

To all of you, I dedicate this work with all my love, gratitude, and sincere appreciation.

Acknowledgement

First and foremost, I would like to thank Almighty God, who gave me the strength, patience, and courage to move forward throughout this journey. It is thanks to Him, above all, that I have reached this point today.

I would also like to express my sincere gratitude to my dear parents for their love, sacrifices, and constant support. Thank you for always believing in me, encouraging me during difficult times, and being my greatest source of strength. I also thank my dear sisters, Meryem and Yasmine, for their presence, affection, and precious support.

I would like to express my deepest gratitude to Mr. MALIKI Fouad , whom I can never thank enough for his help, support, and guidance. My gratitude goes far beyond this work, as throughout these three years, he has always been there for us with exceptional availability, kindness, and dedication. Thanks to his advice, encouragement, and trust, I was able to move forward, learn, and succeed. He was not only a teacher and a head of department, but also a true source of guidance, encouragement, and support throughout our academic journey. I thank him from the bottom of my heart, and I will be grateful to him for the rest of my life.

I would also like to extend my sincere thanks to my supervisor, Mr. ARBAOUI Taha, for his valuable guidance throughout this project. Despite the distance, he was always present through his advice, encouragement, availability, and expertise. His know-how, relevant remarks, and continuous support greatly helped me in the completion of this work. I deeply thank him for everything he has done for me during this project.

I would also like to express my deep appreciation to all the teachers who have accompanied me throughout these years of study. Thank you for your courses, patience, advice, and encouragement, which have contributed to shaping the person I am today. Every piece of knowledge transmitted, every word of guidance, and every effort made has played an important role in my journey. I can never thank them enough for everything they have taught me, and I will always be grateful for every letter, every concept, and every value I have learned from them.

I would also like to thank the members of the jury, Mr. BRAHAMI Mustapha Anwar, Mr. BENNEKROUF Mohammed , and Mr. HADDOU BENDERBAL Hichem , for the honor they have given me by accepting to examine and evaluate this work. I sincerely thank them for the time devoted to reading this thesis, as well as for their remarks and observations, which will undoubtedly contribute to improving and enriching this work.

My thanks also go to my entire family, for their love, prayers, and encouragement, and to my friends, who have always been by my side. Thank you for your support, your listening, your reassuring words, and all the shared moments that made this journey more human, lighter, and more beautiful.

Finally, I would like to thank the little girl I once was, the one who always had the ambition to grow, learn, and reach this point. The one who dreamed in silence, hoped, and believed in a better future, even without always knowing how long and challenging the road would be. Today, this work is also the result of her dreams, perseverance, and faith in what she could become.

To all those who contributed, directly or indirectly, to the completion of this work, I express my sincere thanks and deepest gratitude.

Abstract

This thesis focuses on the optimization of a multi-type human–robot collaborative disassembly line for end-of-life electric vehicle batteries. The studied problem combines several interdependent decisions, including task sequencing, station opening, task-to-station assignment, execution mode selection, resource compatibility, precedence constraints, cycle-time limitations and uncertainty in human operation times.

The objective of this work is to propose a scalable metaheuristic approach capable of solving larger instances for which exact optimization becomes computationally difficult. For this purpose, an Adaptive Large Neighbourhood Search approach is adapted to the structure of the studied problem and denoted HRC-ALNS. Two versions are developed. The deterministic version, called D-HRC-ALNS, considers fixed operation times, while the stochastic version, called S-HRC-ALNS, integrates scenario-dependent human operation times and recourse costs related to station overloads.

The proposed method is based on a specific solution representation composed of a global operation sequence, execution modes and active OR precedence choices. A decoder is then used to reconstruct station assignments, determine resource configurations, compute station loads and evaluate the objective function. Mixed destruction and hybrid repair operators are designed to guide the search toward better solutions. In the stochastic version, additional information such as human-time variability, recourse and worst-scenario loads is also exploited.

Computational experiments were conducted on instances involving one to five battery types. For the small instances, the results were compared with exact MILP solutions. The obtained gaps were very small, confirming the quality of the proposed metaheuristic. For larger instances, the HRC-ALNS was able to generate feasible and interpretable solutions within reasonable computational time. The results show that the proposed approach represents a flexible and efficient solution method for complex human–robot collaborative battery disassembly line problems.

Keywords: Optimization, disassembly line balancing, electric vehicle batteries, human–robot collaboration, ALNS, metaheuristic, stochastic optimization.

Résumé

Ce mémoire porte sur l'optimisation d'une ligne de désassemblage collaborative humain-robot dédiée à plusieurs types de batteries de véhicules électriques en fin de vie. Le problème étudié combine plusieurs décisions interdépendantes, notamment le séquençement des tâches, l'ouverture des stations, l'affectation des tâches aux stations, le choix du mode d'exécution, la compatibilité des ressources, les contraintes de précédence, les limites de temps de cycle et l'incertitude liée aux temps opératoires humains.

L'objectif de ce travail est de proposer une approche métaheuristique capable de traiter des instances de plus grande taille, pour lesquelles la résolution exacte devient difficile en temps de calcul. Pour cela, une méthode basée sur l'Adaptive Large Neighbourhood Search est adaptée à la structure du problème étudié. Cette approche est appelée HRC-ALNS. Deux versions sont développées. La version déterministe, notée D-HRC-ALNS, considère des temps opératoires fixes, tandis que la version stochastique, notée S-HRC-ALNS, prend en compte des temps humains dépendants de scénarios ainsi que des coûts de recours liés aux dépassements de charge des stations.

La méthode proposée repose sur une représentation spécifique de la solution, composée d'une séquence globale des opérations, des modes d'exécution et des choix actifs de précédence OR. Un décodeur permet ensuite de reconstruire les affectations aux stations, de déterminer les configurations de ressources, de calculer les charges des stations et d'évaluer la fonction objectif. Des opérateurs de destruction mixte et de réparation hybride sont conçus afin de guider la recherche vers de meilleures solutions. Dans le cas stochastique, des informations supplémentaires telles que la variabilité des temps humains, les coûts de recours et les charges des scénarios défavorables sont également exploitées.

Les expérimentations numériques ont été réalisées sur des instances comprenant de un à cinq types de batteries. Pour les petites instances, les résultats ont été comparés aux solutions exactes obtenues par MILP. Les écarts obtenus sont très faibles, ce qui confirme la qualité de la métaheuristique proposée. Pour les instances plus grandes, HRC-ALNS a permis de générer des solutions faisables et interprétables dans un temps de calcul raisonnable. Les résultats montrent que l'approche proposée constitue une méthode flexible et efficace pour résoudre des problèmes complexes d'optimisation de lignes de désassemblage collaboratives humain-robot.

Mots-clés : Optimisation, équilibrage de ligne de désassemblage, batteries de véhicules électriques, collaboration humain-robot, ALNS, métaheuristique, optimisation stochastique.

الملخص

يركز هذا العمل على تحسين خط تفكيك تعاوني بين الإنسان والروبوت مخصص لعدة أنواع من بطاريات المركبات الكهربائية في نهاية عمرها. يجمع الشكل المدروس بين عدة قرارات مترابطة، من بينها ترتيب المهام، فتح محطات العمل، إسناد المهام إلى المحطات، اختيار نمط التنفيذ، توافق الموارد، قيود الأسبقية، حدود زمن الدورة، وعدم اليقين المرتبط بأزمة تنفيذ المهام البشرية.

يهدف هذا البحث إلى اقتراح مقارنة ميتاهيورستية قابلة للتوسع من أجل معالجة الحالات ذات الحجم الكبير، التي تصبح فيها طرق الحل الدقيقة مكلفة حسابياً. ولهذا الغرض، تم تكييف خوارزمية البحث التكميلي في الأحياء الكبيرة مع بنية الشكل المدروس، وسميت هذه المقاربة حغثاى ص. تم تطوير نسختين من هذه الطريقة. تعتمد النسخة الحتمية ضحغثاى ص على أزمة تنفيذ ثابتة، بينما تأخذ النسخة العشوائية ضحغثاى ص بعين الاعتبار أزمة تنفيذ بشرية متغيرة حسب السيناريوهات، إضافة إلى تكاليف الرجوع الناتجة عن تجاوز حمولة محطات العمل.

تعتمد الطريقة المقترحة على تمثيل خاص للحل يتكون من تسلسل عام للعمليات، وأنماط التنفيذ، والاختيارات النشطة لعلاقات الأسبقية من نوع اوغ. بعد ذلك، يقوم المفسر بإعادة بناء إسناد المهام إلى المحطات، وتحديد تكوين الموارد، وحساب حمولات المحطات، وتقييم دالة الهدف. كما تم تصميم آليات تدمير مختلطة وإصلاح هجينة لتوجيه البحث نحو حلول أفضل. في الحالة العشوائية، يتم أيضاً استغلال معلومات إضافية مثل تباين أزمة التنفيذ البشرية، وتكاليف الرجوع، وحمولات أسوأ السيناريوهات.

تم اختبار المقاربة المقترحة على حالات تضم من نوع واحد إلى خمسة أنواع من البطاريات. بالنسبة للحالات الصغيرة، تمت مقارنة النتائج مع الحلول الدقيقة المحصل عليها باستعمال البرمجة الخطية المختلطة بالأعداد الصحيحة. أظهرت النتائج أن الفجوات كانت صغيرة جداً، مما يؤكد جودة الميتاهيورستيك المقترح. أما بالنسبة للحالات الأكبر، فقد تمكنت حغثاى ص من إنتاج حلول ممكنة وقابلة للتفسير خلال زمن حسابي معقول. وتبين النتائج أن المقاربة المقترحة تمثل طريقة مرنة وفعالة لحل مشاكل تحسين خطوط التفكيك التعاونية بين الإنسان والروبوت.

الكلمات المفتاحية: التحسين، موازنة خطوط التفكيك، بطاريات المركبات الكهربائية، التعاون بين الإنسان والروبوت، اى ص، الميتاهيورستيك، التحسين العشوائي.

Contents

Abstract	4
Résumé	5
Arabic Abstract	6
List of Figures	11
List of Tables	12
Abbreviations list	12
Abbreviations list	14
General Introduction	15
1 Optimization Methods for Production and Disassembly Systems	17
1.1 Introduction	17
1.2 Fundamentals of Optimization	18
1.2.1 Definition of an Optimization Problem	18
1.2.2 Decision Variables, Objective Functions and Constraints	19
1.2.3 Feasible and Optimal Solutions	20
1.2.4 Optimization as a Decision-Support Process	20
1.3 Classification of Optimization Problems	21
1.3.1 Linear and Nonlinear Optimization	22
1.3.2 Continuous, Integer and Mixed-Integer Optimization	22
1.3.3 Single-Objective and Multi-Objective Optimization	23
1.3.4 Deterministic and Stochastic Optimization	23
1.4 Mathematical Programming	25
1.4.1 Linear Programming	25
1.4.2 Integer Programming	26
1.4.3 Mixed-Integer Linear Programming	26
1.4.4 Relevance to Line Balancing and Scheduling	27
1.5 Optimization in Production and Disassembly Systems	28
1.5.1 Production Line Balancing	29
1.5.2 Assembly and Disassembly Line Balancing	29
1.5.3 Scheduling and Resource Allocation	30
1.5.4 Human–Robot Collaborative Systems	30
1.5.5 Comparison between Production and Disassembly Optimization	31
1.6 Solution Approaches	32
1.6.1 Exact Methods	33
1.6.2 Heuristics	33
1.6.3 Metaheuristics	34

1.6.4	Simulation-Based and Data-Driven Approaches	35
1.6.5	Comparison of Solution Approaches	36
1.7	Optimization under Uncertainty	37
1.7.1	Sources of Uncertainty in Industrial Systems	38
1.7.2	Stochastic Processing Times	39
1.7.3	Robust and Stochastic Optimization Concepts	39
1.8	Link with the Considered Research Problem	40
1.9	Conclusion	41
2	Literature review	43
2.1	Introduction	43
2.2	General Disassembly and Remanufacturing Scheduling Problems	44
2.2.1	Disassembly as a Scheduling and Process Planning Problem	44
2.2.2	Selective, Partial, and Multi-Product Disassembly	45
2.2.3	Exact, Heuristic and Learning-Based Solution Approaches	46
2.3	Electric Vehicle Battery Disassembly: Specific Challenges and Research Trends	46
2.3.1	Specificity of End-of-Life EV Battery Disassembly	47
2.3.2	Manual, Automated, and Human–Robot Assisted Disassembly	47
2.3.3	Main Research Directions in EV Battery Disassembly	48
2.4	Disassembly Sequence Planning for EV Batteries	49
2.4.1	Sequence, Depth, and Circular Strategy Optimization	49
2.4.2	Graph-Based and Matrix-Based Precedence Representation	50
2.4.3	Risk-Aware and Environment-Oriented Sequence Planning	50
2.4.4	Knowledge-Based and AI-Based Sequence Planning	51
2.4.5	Limits of Sequence Planning Approaches	51
2.5	Disassembly Line Balancing for EV Batteries	52
2.5.1	From Sequence Planning to Line Balancing	52
2.5.2	Manual, Robotic, Hybrid and Collaborative Workstations	52
2.5.3	Line Configuration and Operational Performance	53
2.5.4	Cost-Oriented and Performance-Oriented Line Balancing Criteria	53
2.6	Human–Robot Collaboration in Battery Disassembly	54
2.6.1	From Human–Machine Hybrid Systems to Real Collaboration	55
2.6.2	Task Classification in HRC Disassembly	55
2.6.3	Synchronization and Resource Constraints in Collaborative Stations	56
2.6.4	Learning-Based HRC and Remaining Implementation Issues	56
2.7	Uncertainty in Disassembly Planning	57
2.7.1	Sources of Uncertainty in End-of-Life Battery Disassembly	57
2.7.2	Fuzzy, Stochastic, and Scenario-Based Approaches	58
2.7.3	Stochastic Disassembly Line Balancing Models	58
2.7.4	Limits of Existing Stochastic Approaches	59
2.8	Multi-Product Disassembly Lines	59
2.8.1	Motivation for Multi-Product Disassembly	60
2.8.2	Multi-Product Modeling Approaches	60
2.8.3	Destructive Operations, Parallel Layouts and Industrial Extensions	61
2.8.4	Multi-Product HRC and Remaining Limitations	61
2.9	Synthesis of the Literature and Research Gap	62
2.9.1	Main Trends Identified in the Literature	62
2.9.2	Main Limitations	63

2.9.3	Positioning of the Present Research	64
2.10	Conclusion	65
3	Proposed HRC-ALNS Methodology for the Multi-Type Human–Robot Collaborative Battery Disassembly Problem	66
3.1	Introduction	66
3.2	Research Problem and Methodological Positioning	67
3.3	From ALNS to the Proposed HRC-ALNS Variant	68
3.3.1	General Principle of Adaptive Large Neighbourhood Search	68
3.3.2	Motivation for Adapting ALNS to the Studied Problem	69
3.3.3	Definition of the Proposed HRC-ALNS Variants	69
3.4	Solution Representation	70
3.4.1	Internal Encoding of a Solution	71
3.4.2	Encoding of Execution Modes	71
3.4.3	Encoding of OR Precedence Choices	72
3.4.4	Why Stations Are Not Directly Encoded	72
3.5	Initial Solution Construction	73
3.6	Decoding Procedure	73
3.6.1	Precedence-Based Feasibility	74
3.6.2	Station Assignment and Resource Allocation	74
3.6.3	Deterministic Decoding	74
3.6.4	Stochastic Decoding and Recourse Evaluation	75
3.7	Neighbourhood Generation by Mixed Destruction	75
3.7.1	Destruction Size	75
3.7.2	Deterministic Destruction Operators	76
3.7.3	Stochastic Destruction Operators	76
3.7.4	Mixed Destruction Strategy	77
3.8	Hybrid Repair Mechanism	78
3.8.1	Feasible Insertion Positions	78
3.8.2	Deterministic Repair Rules	79
3.8.3	Stochastic Repair Rules	79
3.8.4	Coupling Between Destruction and Repair	80
3.9	Evaluation of Candidate Solutions	81
3.9.1	Deterministic Evaluation	81
3.9.2	Stochastic Evaluation	82
3.9.3	Penalty Management	82
3.10	Acceptance Criterion and Adaptive Operator Selection	82
3.10.1	Simulated Annealing Acceptance Rule	82
3.10.2	Operator Scoring and Weight Updating	83
3.11	General Algorithmic Framework	83
3.12	Conclusion	89
4	Computational Results of the Proposed HRC-ALNS Methodology	91
4.1	Introduction	91
4.2	Experimental Design	91
4.2.1	Implementation Environment	92
4.2.2	Test Instances	92
4.2.3	Main Parameter Settings	92
4.2.4	Stopping Criteria	93

4.2.5	Performance Indicators	94
4.3	Validation of HRC-ALNS Against Exact MILP Results	94
4.3.1	Deterministic Validation	94
4.3.2	Stochastic Validation	95
4.4	Deterministic HRC-ALNS Results	95
4.5	Stochastic HRC-ALNS Results	97
4.6	Comparison and Gap Analysis	98
4.6.1	Deterministic Comparison	98
4.6.2	Stochastic Comparison	98
4.6.3	Interpretation of the Gap Results	99
4.7	Graphical Analysis of the Obtained Schedules	99
4.7.1	Deterministic Gantt Charts	100
4.7.2	Stochastic Gantt Charts	102
4.8	Discussion	104
4.9	Conclusion	105
	General Conclusion	106

List of Figures

1.1	General optimization process for industrial decision problems.	21
1.2	Complementary role of exact optimization and metaheuristic search according to instance size and combinatorial complexity.	35
3.1	Internal encoding of a solution and decoding toward stations.	73
3.2	Example of mixed destruction: several criteria contribute to the removed subset.	78
3.3	Example of hybrid repair after mixed destruction.	81
3.4	General workflow of the proposed HRC-ALNS methodology.	84
4.1	Deterministic Gantt chart obtained with D-HRC-ALNS for the one-battery instance.	100
4.2	Deterministic Gantt chart obtained with D-HRC-ALNS for the two-battery instance.	100
4.3	Deterministic Gantt chart obtained with D-HRC-ALNS for the three-battery instance.	101
4.4	Deterministic Gantt chart obtained with D-HRC-ALNS for the four-battery instance.	101
4.5	Deterministic Gantt chart obtained with D-HRC-ALNS for the five-battery instance.	101
4.6	Stochastic Gantt chart obtained with S-HRC-ALNS for the one-battery instance, scenario 1.	102
4.7	Stochastic Gantt chart obtained with S-HRC-ALNS for the two-battery instance, scenario 1.	102
4.8	Stochastic Gantt chart obtained with S-HRC-ALNS for the three-battery instance, scenario 1.	103
4.9	Stochastic Gantt chart obtained with S-HRC-ALNS for the four-battery instance, scenario 1.	103
4.10	Stochastic Gantt chart obtained with S-HRC-ALNS for the five-battery instance, scenario 1.	103

List of Tables

1.1	Main classifications of optimization problems and relevance to industrial systems.	24
1.2	Relevance of mathematical programming approaches to production and disassembly systems.	28
1.3	Comparison between production and disassembly optimization problems. .	32
1.4	Comparison of solution approaches for industrial optimization problems. .	37
1.5	Sources of uncertainty in production and disassembly systems.	38
1.6	Link between optimization concepts and the considered research problem. .	41
3.1	Main components of the proposed HRC-ALNS methodology.	70
3.2	Deterministic destruction operators used in D-HRC-ALNS.	76
3.3	Stochastic destruction operators used in S-HRC-ALNS.	77
3.4	Deterministic repair rules used in D-HRC-ALNS.	79
3.5	Stochastic repair rules used in S-HRC-ALNS.	79
3.6	Examples of coupling between destruction causes and repair rules.	80
3.7	Main differences between D-HRC-ALNS and S-HRC-ALNS.	89
4.1	Composition of the tested HRC-ALNS instances	92
4.2	Main HRC-ALNS parameter settings	93
4.3	Stopping criteria used in the HRC-ALNS experiments	93
4.4	Validation of D-HRC-ALNS against deterministic MILP results	94
4.5	Validation of S-HRC-ALNS against stochastic MILP results	95
4.6	Deterministic results obtained with D-HRC-ALNS	96
4.7	Stochastic results obtained with S-HRC-ALNS	97
4.8	Deterministic comparison between MILP and D-HRC-ALNS	98
4.9	Stochastic comparison between MILP and S-HRC-ALNS	99

Abbreviations list

Abbreviations list

Acronym	Meaning
AI	Artificial Intelligence
ALNS	Adaptive Large Neighbourhood Search
AND	Mandatory precedence relation requiring all selected predecessors to be completed
CPU	Central Processing Unit
CT	Cycle Time
D-HRC-ALNS	Deterministic Human–Robot Collaborative Adaptive Large Neighbourhood Search
DLBP	Disassembly Line Balancing Problem
EV	Electric Vehicle
H	Human execution mode
HRC	Human–Robot Collaboration
HRC-ALNS	Human–Robot Collaborative Adaptive Large Neighbourhood Search
IP	Integer Programming
LP	Linear Programming
MILP	Mixed-Integer Linear Programming
MOEA/D	Multi-Objective Evolutionary Algorithm based on Decomposition
NSGA-II	Non-dominated Sorting Genetic Algorithm II
OR	Alternative precedence relation where one selected predecessor is sufficient
R	Robotic execution mode
C	Collaborative human–robot execution mode
SA	Simulated Annealing
S-HRC-ALNS	Stochastic Human–Robot Collaborative Adaptive Large Neighbourhood Search

General Introduction

Industrial systems are increasingly faced with complex decisions involving tasks, resources, workstations, processing times, precedence relations, and operational constraints. In production, assembly, disassembly, maintenance, and logistics, decision-makers must determine how operations should be organized, which resources should be used, how workloads should be distributed, and how costs or delays can be reduced. These decisions are strongly connected. Assigning a task to one station, for example, may affect the workload of that station, the availability of human or robotic resources, the feasibility of the cycle time, and the overall performance of the system. This is why optimization plays an important role in industrial engineering.

Optimization makes it possible to translate a practical decision problem into a structured model composed of decision variables, an objective function, and constraints. Through this modelling process, different alternatives can be compared according to clear performance criteria. In line balancing and scheduling problems, this becomes particularly important because the number of possible configurations increases rapidly with the number of tasks, products, stations, and resources. As a result, manual reasoning alone is no longer sufficient, and the use of exact methods, heuristics, or metaheuristics becomes necessary.

Disassembly systems represent a particularly challenging application of industrial optimization. Unlike classical production systems, where products and operations are generally standardized, disassembly systems often deal with end-of-life products whose structure, condition, and accessibility may vary. In the case of electric vehicle batteries, this complexity is even more significant. Battery packs are composed of mechanical, electrical, thermal, and electronic components that must be removed according to strict precedence relations. Some operations are hazardous, some require human flexibility and judgment, while others may be suitable for robotic execution. Therefore, planning a battery disassembly line cannot be limited to defining a task sequence. It must also consider task assignment, workstation configuration, execution modes, resource compatibility, and workload distribution.

Human-robot collaboration provides a promising way to organize such disassembly operations. Human operators bring flexibility, perception, and the ability to adapt to unexpected product conditions. Robots, on the other hand, can support repetitive, precise, heavy, or risky operations. By combining these two types of resources, a collaborative disassembly line can benefit from both human adaptability and robotic assistance. However, this also increases the complexity of the planning problem. Each task may require a human, a robot, or both resources simultaneously. Collaborative tasks must be assigned to stations equipped with both human and robotic resources, while flexible tasks may be executed by either one of them. All these decisions must remain consistent with precedence constraints and cycle-time limitations.

The problem becomes more difficult when several battery types are processed on the same line. Each battery type may have its own task set, processing times, precedence relations, and feasible execution modes. The optimization approach must therefore preserve the specific structure of each product while using a common set of potential stations and resources. This multi-type configuration is closer to the reality of flexible disassembly systems, but it also increases the number of possible decisions and the size of the search space.

Another important aspect is uncertainty. In real disassembly operations, human processing times may vary depending on the actual condition of the battery, the accessibility of components, the operator's adaptation, and unexpected difficulties during task execution. A deterministic model based on fixed operation times can provide a useful baseline, but it does not fully capture this variability. For this reason, a stochastic representation based on scenarios can be used to evaluate the effect of uncertain human operation times on station loads and recourse costs.

Exact optimization methods, such as mixed-integer linear programming, are valuable because they provide rigorous formulations and optimal solutions for small instances. They also make it possible to define the problem clearly and to obtain reference results for validation. However, when the number of battery types, tasks, stations, execution modes, and scenarios increases, the model becomes highly combinatorial. In such cases, exact optimization may require excessive computation time or may become difficult to apply in practice. This motivates the development of a scalable metaheuristic approach capable of producing high-quality solutions within reasonable computational time.

The objective of this work is to develop and evaluate a metaheuristic approach for the multi-type human-robot collaborative battery disassembly line problem. The proposed method is based on Adaptive Large Neighbourhood Search and is adapted to the specific structure of the studied problem. This adaptation is referred to as HRC-ALNS, for Human-Robot Collaborative Adaptive Large Neighbourhood Search. Two versions are considered. The deterministic version, denoted D-HRC-ALNS, uses fixed operation times. The stochastic version, denoted S-HRC-ALNS, considers scenario-dependent human operation times and evaluates recourse costs associated with station overloads.

The proposed methodology relies on an internal solution representation composed of three elements: a global operation sequence, selected execution modes, and active OR precedence choices. Station assignments are not encoded directly. Instead, they are reconstructed by a decoder, which assigns tasks to stations, determines station types, computes station loads, and evaluates the objective function. The search process is based on mixed destruction and hybrid repair operators. In the deterministic case, these operators use information such as bottleneck stations, long tasks, costly tasks, and flexible normal tasks. In the stochastic case, they also exploit recourse values, human-time variability, and worst-scenario information.

The remainder of this thesis is organized as follows. Chapter 1 introduces the main optimization concepts required to understand the methodological choices of the work. It presents optimization problems, mathematical programming, line balancing, scheduling, exact methods, heuristics, metaheuristics, and uncertainty modelling. Chapter 2 reviews the literature related to disassembly planning, electric vehicle battery disassembly, human-robot collaboration, uncertainty, and multi-product disassembly lines. Chapter 3 presents the proposed HRC-ALNS methodology, including the solution encoding, decoding procedure, destruction and repair operators, evaluation functions, and general algorithmic framework. Chapter 4 presents the computational results obtained with the deterministic and stochastic versions of the proposed method. It compares the metaheuristic results with exact MILP references when available and analyses the scalability of the proposed approach on larger instances.

Optimization Methods for Production and Disassembly Systems

1.1 Introduction

Industrial systems are composed of many decisions that must be made under technical, economic, and organizational constraints. In production, assembly, disassembly, maintenance, logistics, and resource planning, decision-makers often have to choose how to allocate resources, how to sequence operations, how to balance workloads, and how to reduce costs or delays. These decisions are rarely independent. A choice made at one station can affect the workload of the next station, the availability of resources, the cycle time of the line, and the global performance of the system. For this reason, optimization has become an important methodological framework for analysing and improving industrial decision problems.

Optimization can be defined as the process of selecting the best solution among a set of feasible alternatives according to one or several performance criteria. In a mathematical formulation, the problem is generally described by decision variables, an objective function, and a set of constraints. The decision variables represent the choices to be made, the objective function expresses what must be minimized or maximized, and the constraints define the technical, physical, operational, or economic limits that must be respected. This structure makes optimization particularly useful for industrial engineering problems, where decisions must be justified and evaluated in a systematic way [42, 8].

In production and disassembly systems, optimization problems often involve discrete decisions. For example, a task may be assigned to one workstation or another, a resource may be selected among several alternatives, a station may be opened or not, and a task may be executed before or after another task according to precedence relations. These characteristics make many industrial problems combinatorial in nature. As the number of tasks, products, stations, and resources increases, the number of possible solutions grows rapidly, making intuitive or manual decision-making insufficient.

Line balancing and scheduling are two important examples of optimization problems in industrial systems. Line balancing focuses on distributing tasks among workstations in order to obtain an efficient workload distribution while respecting cycle time and precedence constraints. Scheduling focuses on the temporal organization of operations, including start times, completion times, and the use of shared resources. In practice, these two aspects are often related: a feasible line configuration must not only assign tasks to stations, but also ensure that the required human, robotic, or collaborative resources are available at the right time.

The importance of optimization becomes even greater in systems that involve product variability and human-robot collaboration. When several product types are processed on the same line, each product may have its own task set, processing times, and precedence relations. When human and robotic resources are used together, the problem must

also consider resource compatibility, task execution modes, synchronization, and safety constraints. In such contexts, optimization provides a structured way to compare alternatives and identify solutions that respect operational constraints while improving cost, workload, productivity, or robustness.

This chapter introduces the optimization concepts required to understand the methodological choices of the present work. It first presents the fundamental elements of an optimization problem, including decision variables, objective functions, constraints, feasible solutions, and optimal solutions. It then classifies optimization problems according to their mathematical structure, decision variable type, number of objectives, and treatment of uncertainty. Mathematical programming methods are introduced with particular attention to linear, integer, and mixed-integer linear programming, since these formulations provide a rigorous way to describe industrial decision problems. The chapter also discusses the role of optimization in production and disassembly systems, including line balancing, scheduling, resource allocation, and human–robot collaboration. Finally, exact methods, heuristics, metaheuristics and uncertainty-related concepts are presented in order to prepare the transition toward the literature review and, later, toward the development of a metaheuristic solution approach for large-scale instances.

1.2 Fundamentals of Optimization

Optimization is a systematic way of supporting decision-making. In many industrial situations, several choices are possible, but not all of them are equally good. Some choices may reduce cost, others may improve productivity, balance workload, reduce idle time, or increase safety. The role of optimization is to describe these alternatives mathematically and to identify the solution that best satisfies the objectives of the system while respecting its constraints.

In industrial engineering, optimization is useful because decisions are usually connected to each other. Assigning a task to one station may influence the load of that station, the use of human or robotic resources, the feasibility of the cycle time, and the availability of resources for other tasks. Similarly, choosing a sequence of operations may affect safety, precedence feasibility, and total completion time. For this reason, optimization provides a structured framework that helps transform a practical decision problem into a mathematical problem that can be analysed and solved.

1.2.1 Definition of an Optimization Problem

An optimization problem consists of finding the best solution from a set of feasible solutions. The term “best” depends on the objective considered. In some cases, the objective is to minimize a cost, a processing time, a number of stations, or a delay. In other cases, the objective is to maximize productivity, profit, resource utilization, or recovered value. The feasible solutions are the solutions that satisfy all the restrictions imposed by the system.

A general optimization problem can be written as follows:

$$\begin{aligned}
 \min \quad & f(x) \\
 \text{s.t.} \quad & g_i(x) \leq 0, \quad i = 1, \dots, m, \\
 & h_j(x) = 0, \quad j = 1, \dots, p, \\
 & x \in X.
 \end{aligned} \tag{1.1}$$

In this formulation, x represents the vector of decision variables, $f(x)$ is the objective function, $g_i(x)$ and $h_j(x)$ represent inequality and equality constraints, and X defines the domain of the decision variables. This formulation is general and can represent many types of problems, from simple continuous optimization to complex mixed-integer models [8, 36].

In the context of production and disassembly systems, the vector x may include decisions such as assigning tasks to stations, selecting execution modes, opening or closing workstations, determining start times, assigning operators or robots, and sequencing tasks. The objective function may represent the minimization of total cost, cycle time, number of stations, idle time, or workload imbalance. The constraints may express precedence relations, resource availability, safety rules, station capacity, and technological feasibility.

1.2.2 Decision Variables, Objective Functions and Constraints

The first element of an optimization model is the set of decision variables. These variables represent the decisions that must be made. They can be continuous, integer, binary, or mixed. A continuous variable can take any value within a given range, such as a processing time or a quantity. An integer variable takes whole-number values, such as the number of stations or the number of operators. A binary variable takes only two values, usually 0 or 1, and is often used to represent yes/no decisions.

For example, in a line balancing problem, a binary variable may indicate whether task i is assigned to station k . In a human–robot collaborative system, another binary variable may indicate whether a task is performed by a human, by a robot, or by both resources. Such variables are useful because many industrial decisions are discrete: a task is either assigned to a station or not, a resource is either selected or not, and a station is either opened or not.

The second element is the objective function. The objective function translates the performance criterion into a mathematical expression. In industrial systems, common objectives include minimizing total cost, minimizing the number of opened stations, minimizing cycle time, minimizing idle time, minimizing energy consumption, or maximizing recovered value. The choice of the objective function is important because it reflects the priorities of the decision-maker.

The third element is the set of constraints. Constraints define what is allowed and what is not allowed. They may come from technical limitations, safety requirements, resource availability, physical layout, precedence relations, or economic restrictions. In a disassembly line, for instance, a component cannot be removed before the elements that block its access have already been removed. This creates precedence constraints. Similarly, a collaborative task cannot be performed unless both a human and a robot are available at the same station. This creates resource compatibility and synchronization constraints.

A good optimization model must therefore balance simplicity and realism. If the model is too simple, it may ignore important industrial constraints and produce solutions that are mathematically optimal but practically infeasible. If the model is too detailed, it may become difficult to solve or require data that are not available. The modelling step is therefore a key part of optimization, because it determines how the real system is represented and what type of solution can be obtained.

1.2.3 Feasible and Optimal Solutions

A feasible solution is a solution that satisfies all constraints of the optimization problem. Feasibility is essential because a solution that violates one or more constraints cannot be implemented in practice. For example, a task assignment that exceeds the cycle time of a station is not feasible. Similarly, a disassembly sequence that removes a module before disconnecting its electrical links is not feasible from a safety and precedence point of view.

The feasible region is the set of all feasible solutions. In simple continuous optimization problems, this region can sometimes be represented geometrically. In industrial combinatorial problems, however, the feasible region is often very large and discrete. It may contain a huge number of possible task assignments, sequences, and resource combinations. This is one of the reasons why optimization problems in production and disassembly systems can become difficult to solve.

An optimal solution is a feasible solution that gives the best value of the objective function. If the objective is minimization, the optimal solution is the feasible solution with the smallest objective value. If the objective is maximization, it is the feasible solution with the largest objective value. In some problems, the optimal solution can be found exactly. In other problems, especially large-scale industrial problems, finding the exact optimum may require significant computational effort. In such cases, approximate methods, heuristics, or metaheuristics may be used to obtain high-quality solutions within a reasonable time.

It is also important to distinguish between local and global optimality. A local optimum is a solution that is better than nearby solutions, but not necessarily better than all feasible solutions. A global optimum is the best solution over the entire feasible region. This distinction is particularly important in nonlinear optimization and metaheuristic methods, where the search process may converge to a good solution without proving that it is globally optimal [36].

In production and disassembly systems, optimality should also be interpreted carefully. A mathematically optimal solution is valuable only if the model correctly represents the real decision problem. If important constraints such as safety, resource compatibility, processing time variability, or human–robot synchronization are ignored, the obtained solution may not be useful in practice. Therefore, optimization must be seen not only as a calculation tool, but also as a modelling and decision-support process.

1.2.4 Optimization as a Decision-Support Process

Optimization is not limited to solving equations. It is a complete decision-support process that starts with understanding the real problem and ends with analysing the obtained solution. The first step is to define the decision context: what must be decided, what objectives are important, and what constraints must be respected. The second step is to formulate the mathematical model. The third step is to choose an appropriate solution method. Finally, the results must be interpreted and evaluated in relation to the real system.

This process is particularly important in industrial applications. A model may produce a solution that is optimal according to the objective function, but the decision-maker must still verify whether the solution is acceptable in practice. For example, a line balancing solution may minimize the number of stations but create an unrealistic workload for human operators. Another solution may reduce cost but increase safety risk or reduce flexibility. Therefore, optimization results should be analysed with both mathematical and practical criteria.

Figure 1.1 summarizes the general optimization process used in industrial decision problems.

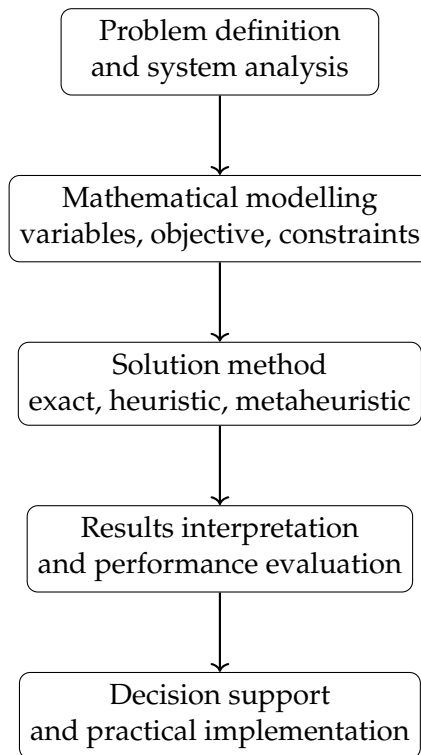


Figure 1.1: General optimization process for industrial decision problems.

In this thesis context, optimization is used as a way to structure complex decisions related to disassembly line organization. The problem involves several interacting elements: multiple battery types, task precedence, station assignment, human and robotic resources, execution modes, and processing time assumptions. These elements cannot be treated independently. Optimization provides a framework for considering them together and for evaluating possible configurations according to well-defined performance criteria.

1.3 Classification of Optimization Problems

Optimization problems can be classified according to different criteria. This classification is important because the mathematical structure of a problem strongly influences the choice of the solution method. A problem with continuous variables is not solved in the same way as a problem with binary assignment decisions. Similarly, a deterministic model does not have the same interpretation as a model that explicitly considers uncertainty. In industrial applications, identifying the type of optimization problem is therefore an essential step before selecting an appropriate formulation and resolution approach.

Production and disassembly systems usually combine several characteristics. They often involve discrete decisions, such as assigning tasks to stations, selecting resources, or opening workstations. They may also include continuous variables, such as start times, completion times, or workload measures. In addition, they can be deterministic when all data are assumed to be known, or stochastic when processing times, product conditions,

or human performance are uncertain. This explains why many industrial optimization problems belong to the family of mixed-integer and combinatorial optimization problems.

1.3.1 Linear and Nonlinear Optimization

A first distinction concerns the mathematical form of the objective function and the constraints. In linear optimization, the objective function and all constraints are linear expressions of the decision variables. This means that variables appear only with constant coefficients, without products between variables, powers, nonlinear functions, or logical nonlinearities. Linear optimization is widely used because many industrial problems can be approximated or formulated in a linear way, and because efficient solution methods exist for this class of problems [8, 42].

A typical linear optimization problem can be written as:

$$\begin{aligned} \min \quad & c^T x \\ \text{s.t.} \quad & Ax \leq b, \\ & x \geq 0, \end{aligned} \tag{1.2}$$

where x is the vector of decision variables, c is the vector of objective coefficients, A is the constraint matrix, and b is the vector of right-hand side values. This formulation is compact, but it can represent many practical situations such as resource allocation, production planning, transportation, and scheduling approximations.

Nonlinear optimization appears when the objective function or at least one constraint is nonlinear. Nonlinearity may come from physical laws, economic functions, energy models, risk measures, learning effects, or interaction terms between variables. Nonlinear models can represent reality more accurately in some cases, but they are generally more difficult to solve. They may have several local optima, and finding a global optimum can be challenging depending on the problem structure [36].

In production and disassembly systems, linear models are often preferred when the goal is to obtain a solvable and interpretable formulation. However, nonlinear elements may appear when modelling fatigue, human performance, risk, ergonomic load, energy consumption, or uncertain behaviour. In such cases, the modeller must decide whether to keep the nonlinear structure or to use linear approximations.

1.3.2 Continuous, Integer and Mixed-Integer Optimization

A second classification concerns the type of decision variables. In continuous optimization, variables can take any value within a given range. This type of variable is useful for modelling quantities, times, flows, or proportions. For example, a continuous variable can represent the start time of a task, the completion time of an operation, or the workload assigned to a station.

In integer optimization, some or all variables must take integer values. This is common in industrial decision problems because many decisions are discrete. A station is either opened or not. A task is assigned to one station and not to another. A robot is selected or not selected. These yes/no decisions are usually represented using binary variables, which are a special case of integer variables.

Mixed-integer optimization combines continuous and integer variables in the same model. It is particularly important for production, assembly, and disassembly systems because these systems often include both assignment decisions and timing decisions. For

example, a mixed-integer model may use binary variables to assign tasks to stations and continuous variables to represent the start and finish times of tasks.

Mixed-integer linear programming, commonly abbreviated as MILP, is one of the most widely used modelling frameworks in industrial optimization. It is powerful because it can represent logical decisions, assignment constraints, capacity limits, precedence relations, and resource selection in a single mathematical formulation. However, MILP problems can become computationally difficult when the number of tasks, products, stations, and resources increases.

1.3.3 Single-Objective and Multi-Objective Optimization

Another important distinction concerns the number of objectives. In single-objective optimization, the problem has one main performance criterion. For example, the objective may be to minimize total cost, minimize the number of stations, minimize cycle time, or minimize total completion time. This type of formulation is useful when one criterion clearly dominates the decision-making process.

However, many industrial problems involve several conflicting objectives. A solution that minimizes cost may not minimize workload imbalance. A solution that reduces the number of stations may increase the workload of human operators. A solution that improves productivity may reduce flexibility or increase safety risk. In such cases, the problem becomes multi-objective.

Multi-objective optimization aims to consider several criteria simultaneously. Instead of producing a single best solution in the classical sense, it often produces a set of efficient or Pareto-optimal solutions. A solution is Pareto-optimal when no objective can be improved without worsening at least one other objective [33]. This concept is useful in industrial systems because decision-makers often need to compare trade-offs rather than select a solution based on one criterion only.

In disassembly line problems, possible objectives may include minimizing cost, minimizing the number of workstations, minimizing idle time, balancing human workload, reducing hazardous operations, or maximizing recovered value. The choice between a single-objective and a multi-objective formulation depends on the purpose of the study and on the level of detail required by the decision-maker.

1.3.4 Deterministic and Stochastic Optimization

A deterministic optimization problem assumes that all input data are known with certainty. Processing times, costs, resource availability, product structure, and demand are considered fixed. Deterministic models are useful because they are easier to formulate, interpret, and solve. They also provide a clear first representation of the decision problem.

However, many industrial systems are affected by uncertainty. Processing times may vary from one operator to another. Product condition may be unknown before inspection. Human execution times may depend on fatigue, experience, task complexity, or unexpected difficulties. In disassembly systems, uncertainty is even more important because end-of-life products may arrive with different degradation levels, damage conditions, or missing information.

Stochastic optimization explicitly considers uncertainty by representing some parameters as random variables or scenarios. Instead of optimizing for a single fixed data set, the model seeks a solution that performs well under uncertainty. This approach is useful

when the variability of the system has a significant impact on performance [9]. For example, stochastic processing times can be used to model the variability of human operations in a human–robot collaborative disassembly line.

Another related approach is robust optimization, which aims to find solutions that remain feasible and effective under a range of possible parameter values. Robust models are useful when probability distributions are difficult to estimate, but uncertainty ranges are known or can be reasonably defined. In practice, deterministic, stochastic, and robust approaches can be seen as complementary modelling choices rather than competing methods.

Table 1.1 summarizes the main classification criteria of optimization problems and their relevance to production and disassembly systems.

Table 1.1: Main classifications of optimization problems and relevance to industrial systems.

Classification criterion	Main categories	Relevance to production and disassembly systems
Mathematical structure	Linear and nonlinear optimization.	Determines whether the model can be solved using linear programming techniques or requires nonlinear methods.
Variable type	Continuous, integer, binary, and mixed-integer optimization.	Represents timing decisions, assignment decisions, resource selection, and station opening decisions.
Number of objectives	Single-objective and multi-objective optimization.	Allows the model to focus on one criterion or to analyse trade-offs between cost, workload, time, and safety.
Treatment of uncertainty	Deterministic, stochastic, and robust optimization.	Determines whether processing times, product condition, or resource performance are treated as fixed or uncertain.
Problem nature	Continuous, combinatorial, or hybrid problems.	Reflects the complexity of task assignment, sequencing, line balancing, and scheduling decisions.

This classification underlines that optimization is not a single method, but a family of modelling approaches. For production and disassembly systems, mixed-integer formulations are particularly important because they can represent discrete task assignments, resource choices, precedence constraints, and station decisions. When uncertainty is introduced, especially in human task execution times or end-of-life product condition, deterministic models may be extended toward stochastic or robust formulations.

1.4 Mathematical Programming

Mathematical programming is one of the main families of optimization methods used in operations research and industrial engineering. It consists of formulating a decision problem as a mathematical model and solving it using appropriate algorithms. A mathematical programming model usually includes decision variables, an objective function, and constraints. The objective function measures the performance of the solution, while the constraints define the feasible region of the problem.

This modelling framework is particularly useful for production and disassembly systems because many industrial decisions can be expressed mathematically. For example, a task can be assigned to a workstation, a resource can be selected, a precedence relation can be imposed, and a cycle time constraint can be respected. Mathematical programming makes it possible to represent these decisions explicitly and to evaluate them according to a clear performance criterion.

1.4.1 Linear Programming

Linear programming, usually abbreviated as LP, is one of the most fundamental forms of mathematical programming. In a linear programming model, the objective function and all constraints are linear expressions of continuous decision variables. This means that the model does not include products of variables, nonlinear functions, or integer restrictions. Despite this apparent simplicity, linear programming can represent many important industrial problems, such as production planning, resource allocation, blending, transportation, and capacity planning [8, 42].

A standard linear programming model can be written as follows:

$$\begin{aligned} \min \quad & \sum_{j=1}^n c_j x_j \\ \text{s.t.} \quad & \sum_{j=1}^n a_{ij} x_j \leq b_i, \quad i = 1, \dots, m, \\ & x_j \geq 0, \quad j = 1, \dots, n. \end{aligned} \tag{1.3}$$

In this formulation, x_j represents the continuous decision variables, c_j are the objective coefficients, a_{ij} are the technological coefficients, and b_i are the available capacities or limits. The objective is to minimize a linear cost function while satisfying all resource or technical constraints.

Linear programming is attractive because it has strong theoretical foundations and efficient solution methods. The simplex method and interior-point methods are among the most common approaches for solving LP problems. In practice, linear programming is also important because many more complex models, including integer and mixed-integer models, rely on LP relaxations during the solution process [8].

However, pure linear programming is not sufficient for many production and disassembly problems. In line balancing, for example, a task cannot be partially assigned to several stations if the operation must be performed as a whole. A workstation is either opened or not. A robot is either installed or not. These decisions require integer or binary variables, which leads to integer programming.

1.4.2 Integer Programming

Integer programming, abbreviated as IP, extends mathematical programming by requiring some or all decision variables to take integer values. When the variables are restricted to 0 or 1, the model is called binary integer programming. Binary variables are especially useful for representing logical decisions in industrial systems.

For example, in a line balancing problem, a binary variable can be defined as follows:

$$x_{ik} = \begin{cases} 1, & \text{if task } i \text{ is assigned to station } k, \\ 0, & \text{otherwise.} \end{cases} \quad (1.4)$$

This type of variable is very useful because it directly represents assignment decisions. Similar binary variables can be used to indicate whether a station is opened, whether a robot is used, whether a task is performed by a human, or whether a specific execution mode is selected.

Integer programming is powerful, but it is generally more difficult to solve than linear programming. The main reason is that the feasible region becomes discrete. Instead of searching over a continuous space, the solver must explore combinations of integer decisions. As the number of binary variables increases, the number of possible combinations can grow very quickly. This phenomenon is one of the sources of computational complexity in line balancing, scheduling, and resource allocation problems [45].

Several exact methods are used to solve integer programming problems. Branch-and-bound is one of the most important methods. It systematically divides the solution space into smaller subproblems and uses bounds to eliminate regions that cannot contain an optimal solution. Branch-and-cut extends this idea by adding valid inequalities, called cuts, to strengthen the formulation and reduce the search space. These methods can prove optimality, but their computational time may become large for difficult industrial problems.

1.4.3 Mixed-Integer Linear Programming

Mixed-integer linear programming, commonly abbreviated as MILP, combines continuous variables and integer or binary variables in a linear model. This makes MILP one of the most important modelling frameworks for industrial optimization. Continuous variables can represent times, workloads, or costs, while binary variables can represent assignment, selection, sequencing, or activation decisions.

A general MILP model can be expressed as:

$$\begin{aligned} \min \quad & c^T x + d^T y \\ \text{s.t.} \quad & Ax + By \leq b, \\ & x \geq 0, \\ & y \in \{0, 1\}^q, \end{aligned} \quad (1.5)$$

where x represents continuous variables and y represents binary variables. The matrices A and B describe the coefficients of continuous and binary variables in the constraints. This structure is flexible and can represent a wide range of industrial decision problems.

MILP is particularly relevant to production and disassembly systems because these systems often combine physical timing decisions and discrete assignment decisions. For example, in a disassembly line, binary variables can assign tasks to stations, select execution modes, or indicate whether a station is used. Continuous variables can represent

starting times, completion times, cycle time, idle time, or workload. Constraints can then enforce precedence relations, resource availability, station capacities, and non-overlap between tasks.

The main advantage of MILP is that it provides a rigorous modelling structure and, when solved exactly, can provide optimal solutions or optimality gaps. This is important when the decision-maker needs to justify the quality of a solution. However, MILP models can become computationally demanding for large-scale problems, especially when they include many products, tasks, stations, resources, and precedence relations.

For this reason, MILP is often used in two ways. In small or medium-sized instances, it can be solved directly using exact optimization methods. In larger instances, it can be used as a reference model to understand the structure of the problem, evaluate heuristic solutions, or provide bounds for approximate methods. This dual role makes MILP useful both as a practical solution tool and as a scientific modelling framework.

1.4.4 Relevance to Line Balancing and Scheduling

Line balancing and scheduling problems are strongly connected to mathematical programming. In line balancing, tasks must be assigned to workstations while respecting precedence relations and cycle time constraints. In scheduling, tasks must be ordered over time while respecting resource availability and technological constraints. When both aspects are considered together, the problem becomes more complex because station assignment and temporal feasibility must be satisfied simultaneously.

In a production or disassembly line, a typical line balancing decision is to determine where each task should be performed. This can be represented using binary assignment variables. The workload of each station is then computed from the processing times of the tasks assigned to it. A cycle time constraint ensures that the workload of each station does not exceed the allowed time. Precedence constraints ensure that tasks are assigned and ordered in a technically feasible way.

In scheduling, additional variables may be needed to represent start times and completion times. If two tasks require the same resource, they cannot be executed simultaneously. This creates non-overlap constraints. In human–robot collaborative systems, the scheduling problem becomes even more detailed because some tasks may require only a human, others only a robot, and some both resources at the same time. Collaborative tasks therefore require synchronization between human and robotic resources.

Table 1.2 summarizes the relevance of LP, IP, and MILP to production and disassembly system modelling.

Table 1.2: Relevance of mathematical programming approaches to production and disassembly systems.

Approach	Main characteristics	Relevance to industrial systems
Linear programming	Continuous variables with linear objective and linear constraints.	Useful for resource allocation, production planning, capacity planning, and LP relaxations of more complex models.
Integer programming	Integer or binary variables used to represent discrete decisions.	Useful for task assignment, station opening, resource selection, and logical decision-making.
Mixed-integer linear programming	Combines continuous and binary/integer variables in a linear formulation.	Suitable for line balancing, scheduling, task allocation, precedence constraints, and human–robot execution mode selection.

For the context of this work, mixed-integer linear programming is especially important. The organization of a disassembly line involves discrete decisions, such as assigning tasks to stations and selecting execution modes, as well as continuous decisions, such as representing processing times or task start times. MILP therefore provides a natural modelling framework for describing such systems in a rigorous way. At the same time, the complexity of MILP models explains why exact methods may need to be complemented by heuristics, metaheuristics, or simulation-based approaches for larger or more uncertain industrial cases.

1.5 Optimization in Production and Disassembly Systems

Optimization plays an important role in the design and management of production and disassembly systems. These systems require several decisions to be made simultaneously, such as assigning tasks to stations, sequencing operations, selecting resources, balancing workloads, and respecting technical constraints. In many cases, these decisions are interdependent. A task assignment decision can influence station workload, resource availability, cycle time feasibility, and the global performance of the line.

Production systems have traditionally been studied from a forward-flow perspective, where raw materials and components are transformed into finished products. Disassembly systems follow the opposite direction: a product is separated into subassemblies, components, or materials for reuse, remanufacturing, recycling, or disposal. Although both systems involve tasks, resources, stations, and precedence relations, disassembly systems usually face additional difficulties because end-of-life products may be heterogeneous, damaged, incomplete, or uncertain in condition.

In this context, optimization provides a structured way to represent production and disassembly decisions. It helps determine how tasks should be distributed, how resources should be used, and how operational constraints should be respected. The following subsections introduce the main optimization concepts used in production and disassembly systems, with particular attention to line balancing, scheduling, resource allocation, and human–robot collaborative systems.

1.5.1 Production Line Balancing

A production line is usually composed of several workstations arranged according to a process flow. Each product follows the line and receives a set of operations at different stations. The objective of line balancing is to distribute tasks among workstations so that the workload is balanced and the line can operate efficiently. This problem is central in industrial engineering because a poorly balanced line can create bottlenecks, idle time, low productivity, and poor resource utilization.

The basic line balancing problem consists of assigning a set of tasks to stations while respecting precedence constraints and a cycle time. Precedence constraints define the technological order between tasks. For example, one operation may have to be completed before another can start. The cycle time represents the maximum allowed workload at each station. If the total processing time assigned to a station exceeds the cycle time, the line configuration is not feasible.

In a simple assembly line balancing problem, the objective is often to minimize the number of stations for a given cycle time or to minimize the cycle time for a given number of stations. More general formulations may include additional constraints, such as parallel stations, mixed-model production, zoning restrictions, ergonomic constraints, resource limitations, or cost minimization [5, 10, 4].

Line balancing is usually a combinatorial optimization problem. If there are many tasks and several possible stations, the number of possible assignments becomes very large. For this reason, exact methods may become computationally expensive for large instances. Heuristics and metaheuristics are often used when the objective is to obtain good solutions within a reasonable computation time.

The concept of line balancing is relevant to disassembly systems because disassembly operations must also be distributed across stations. However, the reverse nature of disassembly introduces additional constraints related to product condition, recovery objectives, hazardous components, and alternative disassembly routes.

1.5.2 Assembly and Disassembly Line Balancing

Assembly and disassembly line balancing share a common structure: both involve a set of tasks, precedence relations, workstations, processing times, and performance objectives. However, the direction and purpose of the process are different. Assembly builds a product from components, while disassembly separates a product into components or materials. This difference creates important modelling implications.

In assembly, the product structure is generally known, and the process is designed for new components under controlled conditions. In disassembly, the product may be used, damaged, incomplete, or partially degraded. The disassembly process may also stop at different levels depending on the recovery objective. For example, a product may be disassembled only to remove a valuable module, or it may be further dismantled for material recovery. This means that disassembly line balancing may involve selective disassembly, partial disassembly, hazardous part removal, and alternative task sequences.

The disassembly line balancing problem can be defined as the assignment of disassembly tasks to workstations while respecting precedence relations and line constraints. Gun-gor and Gupta introduced early formulations of the disassembly line balancing problem and considered the presence of task failures, which is particularly relevant in end-of-life product recovery [18]. Later reviews emphasized that disassembly line balancing can involve several objectives, such as minimizing the number of workstations, balancing idle

time, removing hazardous parts early, or recovering high-demand components as soon as possible [37].

Compared with assembly line balancing, disassembly line balancing is often more uncertain. The exact state of the product may not be known before inspection. Some components may be missing, damaged, blocked, or impossible to remove without destructive operations. Task times may vary depending on product condition and operator behaviour. As a result, disassembly line balancing models may need to incorporate uncertainty, stochastic processing times, alternative routes, or robust decision-making.

For end-of-life electric vehicle batteries, these issues become even more critical. Battery packs differ in architecture, chemistry, cooling systems, joining methods, and safety constraints. In addition, they may contain residual energy and hazardous components. Therefore, the balancing of an EV battery disassembly line must consider not only workload efficiency, but also safety, resource compatibility, and product variability.

1.5.3 Scheduling and Resource Allocation

Scheduling focuses on the temporal organization of tasks. While line balancing decides where tasks are performed, scheduling decides when tasks are performed and in which order they use shared resources. In production and disassembly systems, scheduling is important because tasks may compete for the same human operator, robot, tool, lifting device, or workstation.

A scheduling problem typically involves operations, processing times, resources, and precedence constraints. The objective may be to minimize completion time, lateness, total flow time, idle time, or resource conflicts. In manufacturing systems, scheduling has been widely studied because it directly affects productivity, delivery performance, and resource utilization [38].

In disassembly systems, scheduling is connected to line balancing because a task assignment is not sufficient if the required resources are not available at the right time. For example, two tasks assigned to the same station cannot be performed simultaneously by the same operator. If a robot is required for several tasks, the schedule must ensure that these tasks do not overlap. If a collaborative task requires both a human and a robot, their availability must be synchronized.

Resource allocation concerns the selection and assignment of available resources to tasks. Resources may include human operators, robots, tools, fixtures, workstations, inspection devices, or transport systems. In human–robot collaborative systems, resource allocation is especially important because tasks may have different execution modes. Some tasks can be performed by humans, some by robots, some by either resource, and others require both resources simultaneously.

The combination of scheduling and resource allocation creates a complex optimization problem. It must ensure that task precedence is respected, resources are not overused, and station workloads remain feasible. This is why mathematical programming and heuristic approaches are often used to organize production and disassembly systems.

1.5.4 Human–Robot Collaborative Systems

Human–robot collaborative systems aim to combine the strengths of human operators and robotic systems. Humans are flexible, adaptive, and capable of interpreting unexpected situations. Robots are repeatable, precise, and useful for heavy, repetitive, or hazardous

operations. In industrial systems where product variability is high, collaboration between humans and robots can provide a compromise between flexibility and automation.

From an optimization perspective, human–robot collaboration introduces additional decision dimensions. The model must not only assign tasks to stations, but also determine the execution mode of each task. A task may be performed by a human, by a robot, or by both resources together. The processing time, cost, risk, and feasibility of a task may depend on the selected mode. Therefore, human–robot collaborative systems require a more detailed representation of task-resource compatibility.

Collaborative tasks also introduce synchronization constraints. If a task requires both a human and a robot, both resources must be available at the same station during the same time interval. This is different from a flexible task that can be performed by either a human or a robot. In the collaborative case, the resources are complementary rather than interchangeable. This distinction is important for modelling human–robot disassembly lines.

Safety must also be considered in human–robot collaboration. Even when collaborative robots are designed to operate near humans, the organization of tasks must prevent collisions, unsafe contact, tool interference, and uncontrolled movements. The optimization problem should therefore be consistent with safe interaction, feasible resource assignment, and realistic task sequencing.

In the context of end-of-life EV battery disassembly, human–robot collaboration is particularly relevant because the product is complex and uncertain. Human operators can adapt to irregular conditions, inspect components, and handle delicate operations. Robots can support repetitive unscrewing, lifting, holding, or hazardous operations. Optimization helps determine how these resources should be combined in order to improve line performance while respecting safety and operational constraints.

1.5.5 Comparison between Production and Disassembly Optimization

Although production and disassembly systems share several modelling concepts, they differ in their objectives and constraints. Production optimization usually focuses on assembling products efficiently, reducing cost, improving throughput, and meeting demand. Disassembly optimization focuses on recovering value from end-of-life products, managing uncertainty, reducing risk, and preparing components or materials for downstream recovery routes.

Table 1.3 summarizes the main differences between production and disassembly optimization problems.

Table 1.3: Comparison between production and disassembly optimization problems.

Aspect	Production / assembly systems	Disassembly systems
Process direction	Forward process: components are assembled into a finished product.	Reverse process: an end-of-life product is separated into components or materials.
Product condition	New components with controlled quality and known specifications.	Used products with possible degradation, damage, missing parts, or unknown condition.
Main objective	Improve productivity, reduce cost, meet demand, and balance workload.	Recover value, remove hazardous parts, support reuse, remanufacturing, or recycling.
Task structure	Usually defined by a planned assembly process.	May include selective disassembly, alternative routes, and uncertain feasibility.
Uncertainty	Often related to demand, processing time, or machine availability.	Often related to product condition, task feasibility, component accessibility, and processing time.
Resource use	Human, machine, or robotic resources in relatively controlled conditions.	Human, robotic, and collaborative resources under higher variability and safety constraints.

This comparison explains why disassembly optimization cannot be treated simply as the reverse of assembly optimization. Even if similar mathematical tools can be used, the modelling assumptions must reflect the specific characteristics of end-of-life products. For EV battery disassembly, this means considering product heterogeneity, safety constraints, resource compatibility, human–robot collaboration, and possible processing time variability.

1.6 Solution Approaches

After formulating an optimization problem, the next step is to choose an appropriate solution approach. This choice is important because the quality of the results does not depend only on the model, but also on the method used to solve it. A small and well-structured problem may be solved exactly, while a large industrial problem may require approximate methods to obtain a good solution within a reasonable computation time.

In production and disassembly systems, solution approaches must deal with several sources of complexity: task assignment, precedence relations, station capacity, resource compatibility, processing times, and sometimes uncertainty. When human–robot collaboration is considered, the problem becomes even more complex because each task may have different feasible execution modes and because collaborative tasks may require synchronization between resources. For this reason, several families of solution methods are commonly used, including exact methods, heuristics, metaheuristics, and simulation-based or data-driven approaches.

1.6.1 Exact Methods

Exact methods aim to find an optimal solution and, when possible, prove that no better feasible solution exists. These methods are particularly useful when the problem size is moderate or when the decision-maker needs a guaranteed optimal solution. In mathematical programming, exact methods are often used to solve linear programming, integer programming, and mixed-integer linear programming models.

For linear programming problems, methods such as the simplex algorithm and interior-point methods are widely used. For integer and mixed-integer problems, branch-and-bound and branch-and-cut are among the most important exact approaches. Branch-and-bound explores the solution space by dividing it into smaller subproblems and using bounds to eliminate parts of the search space that cannot contain an optimal solution. Branch-and-cut strengthens this process by adding valid inequalities to improve the formulation and reduce the number of explored nodes [45, 8].

The main advantage of exact methods is their ability to provide strong guarantees. If the method reaches optimality, the obtained solution is mathematically proven to be optimal for the formulated model. This is useful when the model is used as a benchmark or when the problem is small enough to be solved within an acceptable time.

However, exact methods may become computationally expensive for large combinatorial problems. In line balancing and scheduling, the number of possible assignments and sequences can increase very rapidly with the number of tasks, stations, products, and resources. As a result, exact methods may require a long computation time, especially when the model includes binary variables, precedence constraints, resource conflicts, and synchronization constraints.

For this reason, exact methods are often used in two complementary ways. First, they can solve small and medium-sized instances directly. Second, they can provide reference solutions, lower bounds, or optimality gaps that help evaluate the performance of approximate methods.

1.6.2 Heuristics

Heuristics are solution methods designed to find good feasible solutions without necessarily proving optimality. They are usually based on problem-specific rules, priority criteria, or constructive procedures. In industrial applications, heuristics are valuable because they are often simple, fast, and easier to implement than complex exact algorithms.

A constructive heuristic builds a solution step by step. For example, in a line balancing problem, tasks can be assigned to stations according to priority rules such as longest processing time, largest number of successors, positional weight, or earliest feasible assignment. The objective is not to explore all possible solutions, but to produce a feasible and acceptable solution quickly.

Improvement heuristics start from an initial solution and try to improve it by making local changes. For example, a task may be moved from one station to another, two tasks may be exchanged, or a sequence may be modified. If the change improves the objective function and keeps the solution feasible, it can be accepted. This type of approach is often useful when a first solution is available but still contains workload imbalance, idle time, or inefficient resource use.

The main advantage of heuristics is their computational efficiency. They can handle larger instances than exact methods and can provide solutions quickly. This is important in industrial contexts where decisions must sometimes be made with limited time and incomplete information. The main limitation is that heuristics do not usually provide a

guarantee of optimality. A heuristic solution may be good, but it is not always possible to know how far it is from the best possible solution.

For production and disassembly systems, heuristics are often attractive because they can incorporate practical rules that are difficult to represent in a compact mathematical model. However, their performance depends strongly on the quality of the rules used and on the structure of the problem.

1.6.3 Metaheuristics

Metaheuristics are general-purpose search strategies designed to explore complex solution spaces. Unlike simple heuristics, they are not limited to one fixed construction rule. They rely on mechanisms that alternate between exploration, which searches new regions of the solution space, and exploitation, which improves promising solutions already found. This balance is important in industrial optimization because large combinatorial problems may contain many local optima and many feasible configurations with similar objective values.

Common metaheuristics include genetic algorithms, simulated annealing, tabu search, particle swarm optimization, ant colony optimization, and variable neighbourhood search. Each method follows a different search logic. Genetic algorithms are inspired by evolutionary mechanisms such as selection, crossover, and mutation. Simulated annealing accepts some worsening moves in order to escape local optima. Tabu search uses memory structures to avoid cycling and guide the search toward promising regions. Ant colony optimization is based on the collective behaviour of artificial ants and pheromone-like information [17, 28, 24, 13].

Metaheuristics are useful for line balancing, scheduling, and disassembly problems because these problems often have a combinatorial structure. A solution may involve many simultaneous decisions: assigning tasks to stations, selecting resources, ordering operations, and respecting precedence constraints. Searching this space with exact methods can be difficult when the instance becomes large. Metaheuristics provide a practical alternative by producing good solutions without requiring complete enumeration.

The main advantage of metaheuristics is their flexibility. They can be adapted to different objective functions, constraints, and industrial contexts. They can also be combined with local search, repair mechanisms, or mathematical programming components. Their main limitation is the absence of a general proof of optimality. Their performance depends on the encoding of the solution, the design of neighbourhood operators, the choice of acceptance criteria, the stopping rules, and the tuning of parameters.

In human–robot collaborative disassembly, metaheuristics are particularly relevant because the problem includes both assignment and scheduling decisions. A candidate solution may need to specify the station of each task, the selected execution mode, the resources installed at each station, and the temporal order of operations. These decisions interact strongly with precedence and capacity constraints. A metaheuristic can explore such combinations more flexibly than a simple constructive rule, while requiring less computation time than a complete enumeration of all feasible assignments.

Figure 1.2 illustrates the complementary role of exact optimization and metaheuristic search according to problem size and combinatorial complexity. Exact methods, such as mixed-integer linear programming, are particularly suitable for small instances because they provide optimal solutions and strong mathematical guarantees. However, as the size and complexity of the problem increase, their computational cost becomes much higher. In such cases, metaheuristic approaches become more appropriate, since they are able to

produce good-quality solutions within reasonable computational times. For medium-size instances, both families of methods may play a complementary role, with exact models serving as reference formulations and metaheuristics providing a more scalable search framework.

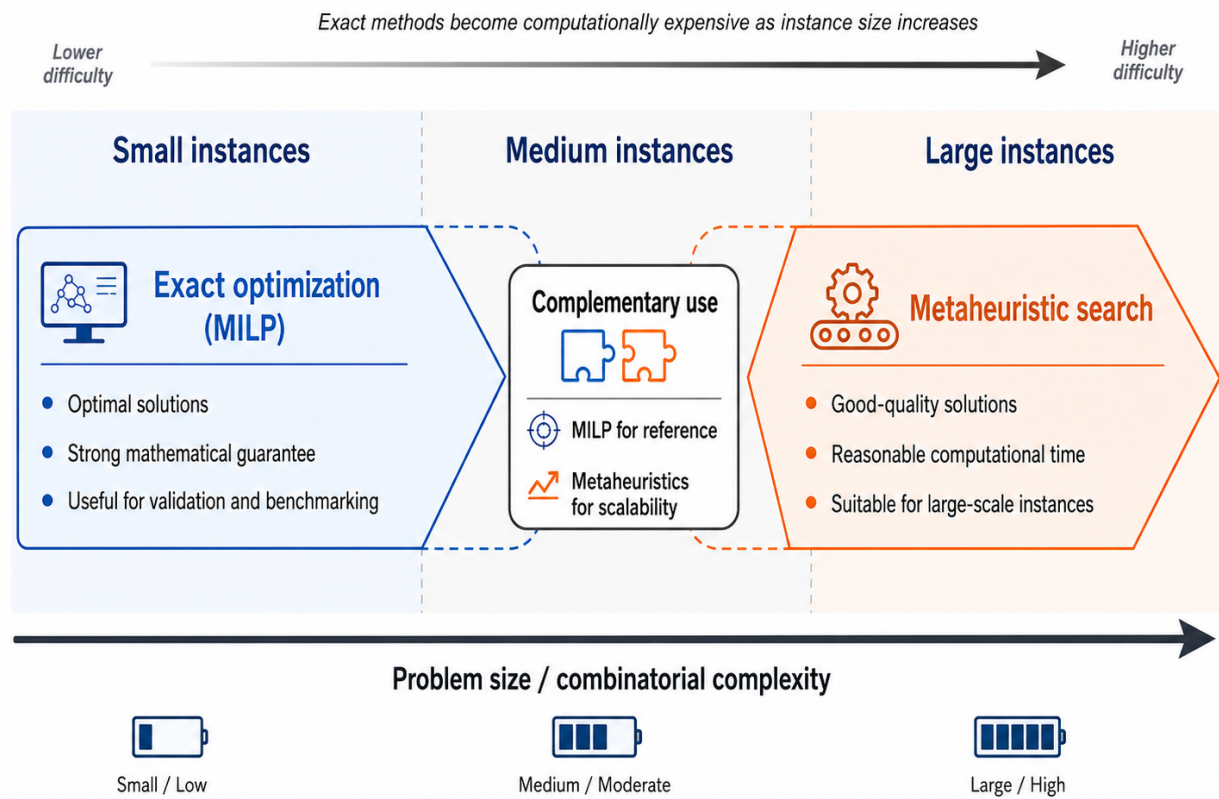


Figure 1.2: Complementary role of exact optimization and metaheuristic search according to instance size and combinatorial complexity.

1.6.4 Simulation-Based and Data-Driven Approaches

Simulation-based approaches are useful when the system contains dynamic behaviour, uncertainty, or interactions that are difficult to capture in a deterministic mathematical model. Instead of only solving equations, simulation reproduces the behaviour of the system under different scenarios. It can be used to evaluate waiting times, resource utilization, bottlenecks, stochastic processing times, and the effect of product variability.

In production and disassembly systems, discrete-event simulation is particularly relevant. It represents the system as a sequence of events, such as the arrival of a product, the start of a task, the completion of an operation, or the release of a resource. This allows the modeller to analyse how the system behaves over time. Simulation does not automatically find the best solution, but it can evaluate and compare different configurations.

Simulation can also be combined with optimization. In simulation-based optimization, a search algorithm proposes candidate solutions, and simulation is used to evaluate their performance. This approach is useful when analytical evaluation is difficult or when the performance depends on random events. For example, if human processing times are

stochastic, simulation can estimate the expected workload, waiting time, or probability of station overload under different task assignments.

Data-driven approaches are increasingly used in industrial optimization. Historical data, sensor information, operator performance data, and product inspection data can help estimate processing times, failure probabilities, resource availability, or uncertainty distributions. In some cases, machine learning can be used to predict task durations or classify product conditions before optimization. However, data-driven methods require reliable data and careful validation before being used for decision-making.

For end-of-life battery disassembly, simulation-based and data-driven methods can be useful because the process is affected by product condition, human variability, and uncertain task durations. A deterministic optimization model may provide a first structured solution, while simulation or data-driven analysis can be used to test how robust this solution is under more realistic operating conditions.

1.6.5 Comparison of Solution Approaches

The choice of a solution approach depends on the size of the problem, the required solution quality, the available computation time, and the level of uncertainty. Exact methods are useful when optimality is required and the instance size is manageable. Heuristics are suitable when fast feasible solutions are needed. Metaheuristics are useful for large and complex problems where good solutions are required but exact optimality is difficult to prove. Simulation-based and data-driven approaches are useful when uncertainty, dynamic behaviour, or empirical data must be considered.

Table 1.4 summarizes the main characteristics of these solution approaches.

Table 1.4: Comparison of solution approaches for industrial optimization problems.

Approach	Main principle	Advantages	Limitations
Exact methods	Search for an optimal solution and may prove optimality.	Provide optimality guarantees or optimality gaps; useful for benchmark instances.	Can become computationally expensive for large combinatorial problems.
Heuristics	Use problem-specific rules to build or improve feasible solutions.	Fast, simple, and easy to adapt to practical constraints.	Usually no guarantee of optimality; performance depends on the rule design.
Metaheuristics	Use general search mechanisms to explore large solution spaces.	Flexible and suitable for complex assignment, sequencing, and scheduling problems.	Require parameter tuning; usually no proof of optimality.
Simulation-based approaches	Evaluate system behaviour under dynamic or uncertain conditions.	Useful for stochastic processing times, resource interactions, and bottleneck analysis.	Does not directly optimize unless combined with a search method.
Data-driven approaches	Use data to estimate parameters, predict outcomes, or support optimization.	Can improve realism when reliable operational data are available.	Depend on data quality, representativeness, and validation.

In practice, these approaches are not mutually exclusive. A mathematical programming model can be solved exactly for small instances and then used as a reference for heuristics or metaheuristics. A heuristic can provide an initial solution to a metaheuristic. A simulation model can evaluate solutions produced by an optimization algorithm. This combination of methods is often useful in industrial systems because it allows the modeller to balance mathematical rigor, computational efficiency, and practical realism.

1.7 Optimization under Uncertainty

Many optimization models begin with a simplifying assumption: all input data are known and fixed. This assumption is useful because it makes the first formulation clearer and easier to solve. In real industrial systems, however, data are rarely perfectly stable. Processing times may vary, resources may not always be available, product conditions may differ, and human performance may change from one task to another. These variations can affect the feasibility and quality of a solution.

Uncertainty is especially important in disassembly systems. Unlike assembly, where components are usually new and standardized, disassembly often deals with products that have been used, degraded, repaired, damaged, or stored in different conditions. For end-of-life batteries, this uncertainty is even more sensitive because the condition of the

pack can influence safety, task duration, accessibility, and the feasible recovery route. A line organization that works well for one battery may become less efficient or even infeasible when another battery has a different condition.

1.7.1 Sources of Uncertainty in Industrial Systems

Uncertainty can appear at several levels of an industrial system. At the product level, the exact structure or condition of the product may not be fully known before inspection. At the process level, operation times may vary because of task complexity, tool condition, or resource performance. At the resource level, human operators and machines may not behave in a perfectly repeatable way. At the system level, demand, product mix, arrivals, and availability of resources can change over time.

In production systems, uncertainty is often related to demand fluctuations, machine breakdowns, material shortages, or processing time variability. In disassembly systems, the uncertainty is broader because the product itself can be uncertain. A component may be damaged, blocked, missing, or difficult to access. Some operations may require additional inspection or adaptation. Therefore, uncertainty is not only an external disturbance; it is part of the disassembly process.

In human–robot collaborative systems, human execution time is an important source of variability. A human operator may perform the same task faster or slower depending on experience, fatigue, task visibility, ergonomic conditions, or unexpected difficulties. This does not mean that human work is unreliable; it means that human work is adaptive and sensitive to context. A realistic optimization approach should therefore be able to represent, or at least acknowledge, this variability.

Table 1.5 summarizes common sources of uncertainty and their possible impact on production and disassembly systems.

Table 1.5: Sources of uncertainty in production and disassembly systems.

Source of uncertainty	Examples	Possible impact
Product condition	Damage, corrosion, deformation, missing information, unknown usage history.	Changes task feasibility, required precautions, disassembly route, and processing time.
Processing time	Variability in manual tasks, tool use, access difficulty, or component removal.	Affects station workload, cycle time feasibility, and schedule reliability.
Resource availability	Operator absence, robot downtime, tool unavailability, maintenance operations.	Can create delays, bottlenecks, or reassignment needs.
Human performance	Experience, fatigue, learning effects, ergonomic conditions, adaptation to unexpected cases.	Influences task duration, quality, and variability of manual operations.
Product mix and arrivals	Different product types arriving with different frequencies and conditions.	Changes workload distribution and required resource capacities.

Recognizing these uncertainties is important before choosing a modelling approach. A deterministic model may be sufficient for a first analysis or when variability is limited. When uncertainty has a strong effect on performance, stochastic or robust modelling can provide more realistic support for decision-making.

1.7.2 Stochastic Processing Times

Processing time is one of the most important parameters in line balancing and scheduling. It determines station workload, resource use, completion times, and bottlenecks. In many models, processing times are assumed to be deterministic. This means that each task has a fixed duration known in advance. This assumption is useful, but it can be too restrictive when tasks are performed by humans or when the product condition is uncertain.

A stochastic processing time is a task duration represented as a random variable. Instead of assuming that a task always takes the same time, the model recognizes that the duration may vary according to a probability distribution or a set of scenarios. For example, a manual disconnection task may usually take three minutes, but it may take longer if a connector is blocked, if access is difficult, or if the operator needs to inspect the component more carefully.

In a human–robot collaborative disassembly line, stochastic processing times may be particularly relevant for human tasks. Robotic tasks can also vary, but they are generally more repeatable when the operation is well defined. Human tasks, on the other hand, often include perception, judgement, adaptation, and delicate handling. These qualities are valuable, but they naturally introduce variability in operation times.

Stochastic processing times can be handled in different ways. A model may use expected processing times, safety margins, scenarios, probability distributions, or chance constraints. Using expected values is simple, but it may underestimate the risk of overload at a station. Adding safety margins can improve robustness, but may reduce line utilization. Scenario-based models represent several possible realizations of processing times and search for solutions that perform acceptably across them [9, 41].

The choice of representation depends on the available data. If historical observations are available, distributions or empirical scenarios can be estimated. If data are limited, the modeller may use expert judgement, intervals, or sensitivity analysis. In all cases, the objective is not to make the model unnecessarily complicated, but to avoid ignoring variability that has a real operational impact.

1.7.3 Robust and Stochastic Optimization Concepts

Stochastic optimization and robust optimization are two common approaches for dealing with uncertainty. They are related, but they do not use the same logic. Stochastic optimization assumes that uncertainty can be described using probability distributions or scenarios. The goal is often to optimize an expected performance measure or to satisfy constraints with a certain probability. This approach is useful when enough information is available to describe the uncertain parameters statistically [9, 41].

Robust optimization, on the other hand, does not require exact probability distributions. Instead, uncertain parameters are assumed to belong to an uncertainty set. The goal is to find a solution that remains feasible and performs reasonably well for all, or most, values inside this set. Robust optimization is useful when probability distributions are difficult to estimate but reasonable bounds on uncertainty are available [7].

Both approaches involve a trade-off. A deterministic model may produce a solution that is efficient for nominal data, but sensitive to deviations. A robust solution may be more reliable, but it can be more conservative. A stochastic solution may represent uncertainty more accurately, but it requires more data and can be harder to solve. Therefore, the choice depends on the objective of the study, the quality of available data, and the level of risk the decision-maker is willing to accept.

For production and disassembly systems, uncertainty can be considered at different modelling levels. At a simple level, sensitivity analysis can be used to test how the solution changes when processing times or product mix vary. At a more advanced level, stochastic or robust formulations can be developed to integrate uncertainty directly into the optimization model. In human-robot disassembly, this is especially relevant when human processing times are variable and when end-of-life products do not arrive in a predictable condition.

1.8 Link with the Considered Research Problem

The optimization concepts presented in this chapter provide the methodological basis for the research problem considered in this work. The problem concerns the organization of an end-of-life electric vehicle battery disassembly line where several battery types may be processed and where human and robotic resources may collaborate. This context combines several features that are typical of complex industrial optimization problems.

First, the problem includes task assignment decisions. Each battery type can be decomposed into a set of disassembly tasks, and these tasks must be assigned to workstations. This is directly related to line balancing because the workload of each station must remain compatible with the cycle time. If the workload is poorly distributed, the line may suffer from bottlenecks or idle resources.

Second, the problem includes precedence constraints. Disassembly tasks cannot be performed in any order because the physical structure of the battery creates accessibility and safety dependencies. Covers, connectors, busbars, cooling elements, and modules must be handled according to a feasible sequence. This combinatorial structure requires a clear representation of task relations.

Third, the problem includes resource allocation. Some tasks are better suited to human operators, others to robots, and some may require collaboration between both. This creates compatibility constraints between tasks and execution modes. A collaborative task is not equivalent to a task that can be performed by either a human or a robot; it requires both resources at the same time, which creates synchronization requirements.

Fourth, the problem may include uncertainty. Human execution times may vary, and end-of-life batteries may arrive with different conditions. A deterministic formulation can provide a first structured representation of the problem, while stochastic or robust extensions can be used to study the influence of processing time variability and uncertainty. This is particularly relevant when human tasks represent a significant part of the disassembly process.

Table 1.6 summarizes the link between the optimization concepts introduced in this chapter and the considered research problem.

Table 1.6: Link between optimization concepts and the considered research problem.

Optimization concept	General meaning	Role in the considered problem
Decision variables	Mathematical representation of decisions to be made.	Assignment of tasks to stations, selection of resources, execution modes, and timing decisions.
Objective function	Performance criterion to minimize or maximize.	Cost, number of stations, workload balance, cycle time, or other performance measures.
Constraints	Conditions that define feasible solutions.	Precedence relations, cycle time, resource compatibility, safety requirements, and non-overlap constraints.
Line balancing	Distribution of tasks among stations.	Organization of multi-type battery disassembly tasks across a balancing line.
Scheduling	Temporal organization of tasks and resources.	Start times, finish times, resource conflicts, and synchronization of human–robot collaborative tasks.
Uncertainty	Variability or lack of perfect knowledge in input data.	Human processing time variability and uncertain end-of-life battery condition.

This connection indicates that the research problem is not only a technical problem of battery disassembly. It is also an optimization problem involving task allocation, precedence, resource compatibility, human–robot collaboration, and possibly uncertainty. The next chapter will build on this methodological foundation to review the literature related to disassembly line balancing, human–robot collaboration, multi-product systems, and stochastic or uncertain processing times.

1.9 Conclusion

This chapter introduced the main optimization concepts needed to study production and disassembly systems. It first defined optimization as a decision-support process based on decision variables, objective functions, constraints, feasible solutions, and optimal solutions. This structure is important because it makes it possible to transform an industrial decision problem into a mathematical model that can be analysed and solved.

The chapter then presented the main classifications of optimization problems. Linear and nonlinear optimization differ by the mathematical form of the objective function and constraints. Continuous, integer, and mixed-integer optimization differ by the type of decision variables. Single-objective and multi-objective optimization differ by the number of performance criteria considered. Deterministic, stochastic, and robust optimization differ by the way uncertainty is represented. These classifications help identify which type of formulation is appropriate for a given industrial problem.

Mathematical programming was then discussed as a central modelling framework. Linear programming, integer programming, and mixed-integer linear programming provide useful tools for representing resource allocation, task assignment, station opening, precedence relations, and scheduling decisions. In production and disassembly systems, mixed-integer formulations are particularly relevant because they can combine discrete assignment decisions with continuous timing or workload variables.

The chapter also explained how optimization is used in production and disassembly systems. Line balancing distributes tasks among workstations, scheduling organizes tasks over time, and resource allocation assigns humans, robots, tools, or other resources to operations. Disassembly systems share some similarities with assembly systems, but they also introduce additional difficulties such as product condition uncertainty, selective disassembly, hazardous components, and alternative recovery routes.

Different solution approaches were presented, including exact methods, heuristics, metaheuristics, simulation-based methods, and data-driven approaches. Each family has its own strengths and limitations. Exact methods can prove optimality but may become computationally expensive. Heuristics and metaheuristics can handle larger or more complex problems but usually do not guarantee optimality. Simulation and data-driven approaches can improve realism when uncertainty and operational data are important.

Finally, the chapter discussed optimization under uncertainty. This aspect is particularly relevant for disassembly systems because end-of-life products are not always predictable and because human processing times may vary. Stochastic and robust optimization provide possible ways to represent uncertainty and to design solutions that remain effective under variable conditions.

The concepts introduced in this chapter provide the methodological basis for the rest of the work. They explain why the organization of a multi-type human–robot battery disassembly line can be studied as an optimization problem and why exact models may need to be complemented by metaheuristic search for larger cases. The following chapter presents the literature review related to disassembly line balancing, human–robot collaboration, multi-product systems, uncertainty, and metaheuristic optimization for end-of-life product recovery.

Literature review

2.1 Introduction

The previous chapter introduced the main families of optimization methods and showed why complex industrial planning problems often require both rigorous mathematical formulations and scalable solution approaches. Building on this methodological background, the present chapter focuses on the scientific literature related to end-of-life electric vehicle battery disassembly. In this context, disassembly is not only a technical operation, but also a strategic activity that enables reuse, remanufacturing, repurposing and recycling. However, the organization of battery disassembly remains difficult because of the high variability of battery architectures, the presence of hazardous components, the lack of standardization, and the uncertainty related to the real condition of end-of-life products.

For this reason, the scientific literature has increasingly focused on the development of decision-support models for disassembly planning. These models aim to determine how a product should be disassembled, in which order the operations should be performed, which resources should be used, and how the workload should be distributed among workstations. In the broader field of remanufacturing and recycling, disassembly has been studied through several related problems, such as disassembly scheduling, selective disassembly planning, disassembly sequence planning, and disassembly line balancing. These problems generally involve precedence constraints, resource capacity constraints, task assignment decisions, and performance criteria related to time, cost, profit, energy or risk.

In the specific case of electric vehicle batteries, the problem becomes even more complex. Battery packs and modules contain electrical, mechanical and thermal components that must be removed in a safe and structured way. Some operations are dangerous because of high voltage or thermal risks, while others require flexibility and human judgment due to product degradation or design heterogeneity. Villagrossi and Dinon [44] showed that robotic systems can support battery pack disassembly, but that product variability still limits full automation. Rettenmeier et al. [40] analyzed disassembly technologies as a key element of the circular battery value chain. Kaarlela et al. [25] reviewed robotised battery disassembly and emphasized the need for flexible tooling and product information. Li et al. [29] also highlighted the role of artificial intelligence and human-robot collaboration in future end-of-life battery disassembly systems.

This chapter presents a review of the main research contributions related to disassembly planning and line balancing, with a progressive focus on end-of-life electric vehicle batteries. The review first discusses general disassembly and remanufacturing scheduling problems, then examines the specific challenges of battery disassembly. After that, it analyzes works on disassembly sequence planning, disassembly line balancing, human-robot collaboration, uncertainty modeling, and multi-product disassembly lines. Particular attention is given to studies that are close to the present research, especially those dealing

with human and robotic resources, stochastic operation times, and several product types.

The purpose of this state of the art is not only to summarize existing approaches, but also to identify the remaining research gaps. The literature shows that many contributions address battery disassembly through sequence optimization, line balancing, or human–robot task allocation. Nevertheless, most works remain focused on a single battery type or rely on deterministic data. Moreover, although uncertainty, multi-product disassembly and human–robot collaboration have been studied separately, their simultaneous integration remains limited. These observations justify the modelling framework on which this work is based: a deterministic mono-objective formulation provides a rigorous reference for the multi-product human–robot collaborative disassembly line, while a stochastic scenario-based extension accounts for uncertainty in human operation times. The computational limits of such exact formulations then motivate the development of a more scalable solution approach for larger instances.

2.2 General Disassembly and Remanufacturing Scheduling Problems

Before focusing on electric vehicle batteries, it is necessary to place the disassembly problem within the broader context of remanufacturing and end-of-life product recovery. In closed-loop supply chains, disassembly is generally the first operational step that makes it possible to recover components, materials, or subassemblies. Unlike classical manufacturing or assembly, where the product structure is usually known and controlled, disassembly deals with returned products whose condition may vary from one unit to another. This makes the planning problem more difficult, especially when only some parts have to be recovered or when several recovery options are possible.

In the literature, disassembly has been studied from different decision-making perspectives. Some works focus on disassembly sequence planning, where the main question is to determine the order in which components should be removed. Other works consider disassembly scheduling, where operations must be assigned to machines or workstations over time. A third stream addresses disassembly line balancing, where tasks are distributed among stations while respecting precedence relations and cycle time constraints. These problems share a common structure: they involve a product precedence graph, a set of tasks or operations, limited resources, and one or several performance criteria.

2.2.1 Disassembly as a Scheduling and Process Planning Problem

In remanufacturing systems, disassembly is rarely treated as an isolated operation. It is often integrated with downstream processes such as cleaning, inspection, reprocessing, and reassembly. For this reason, several authors have formulated integrated planning and scheduling models for remanufacturing systems. In these models, the system usually includes a disassembly stage, a reprocessing stage, and a reassembly stage. The main decisions concern the selection of process routes, the assignment of operations to machines, and the sequencing of products or components.

Early contributions already showed that end-of-life recovery processes can be modeled as scheduling problems. Cheng et al. proposed a scheduling model for refurbishing processes in recycling management, showing that the sequencing of returned products and the weighting of completion times can influence the performance of recovery operations [11]. Although this work does not focus on batteries, it is useful because it introduces

the idea that disassembly and refurbishment are not only technical processes, but also scheduling problems that require optimization.

More recent works integrate process planning and scheduling decisions in remanufacturing environments. Zhang et al. [54] studied an integrated process planning and scheduling problem in a flexible job-shop-type remanufacturing system composed of disassembly, reprocessing and reassembly workshops. In this type of system, the model must determine the assignment and sequence of products in disassembly and reassembly stages, while also selecting processing routes and machines for the reprocessing stage. Zhang et al. [55] extended this logic by considering energy-oriented integrated process planning and scheduling. Another work by Zhang et al. [56] addressed a flexible job-shop-type remanufacturing system where total energy consumption and total completion time are considered.

These studies show that the disassembly stage cannot be completely separated from the rest of the remanufacturing system when the objective is to organize the whole recovery process. However, they mainly focus on general remanufacturing systems and do not always consider the specific constraints of battery disassembly, such as electrical hazards, thermal risks, and the need for human–robot cooperation.

2.2.2 Selective, Partial, and Multi-Product Disassembly

Another important research direction concerns selective and partial disassembly. In many end-of-life products, it is not always economically or technically necessary to remove all components. Instead, the decision-maker may target only valuable, reusable, hazardous, or legally required parts. This leads to selective disassembly planning, where the model must decide which operations are necessary and which can be avoided.

Ehm considered a selective disassembly planning and scheduling problem for multiple products in a job-shop environment. The proposed problem integrates the selection of disassembly operations, the choice between destructive and non-destructive modes, and the scheduling of operations on machines. The model aims to minimize the makespan and environmental impact, while respecting precedence relations, machine capacities, recovery requirements and restrictions related to destructive operations [14]. This work is important because it shows that multi-product disassembly can be formulated in a unified scheduling framework, even when different products have different structures and recovery requirements.

Partial disassembly has also been studied in robotic or automated line configurations. In this case, only demanded parts and hazardous parts may be imposed as mandatory, while other components can be left inside the product if their recovery value is not sufficient. This idea is particularly relevant for industrial recovery systems, because complete disassembly may increase the processing time and cost without necessarily improving the economic result. Therefore, partial disassembly introduces a trade-off between recovery value and operational effort.

For the present research, this literature is useful for two reasons. First, it shows that disassembly planning can be formulated as a structured optimization problem with task selection, precedence constraints and resource assignment. Second, it confirms that the multi-product dimension is important in realistic end-of-life systems. Nevertheless, most multi-product studies remain general and are not specifically designed for electric vehicle batteries or for human–robot collaborative stations.

2.2.3 Exact, Heuristic and Learning-Based Solution Approaches

The problems described above are generally combinatorial and difficult to solve for large instances. Exact methods such as mixed-integer linear programming are useful because they provide a clear mathematical description of the problem and can obtain optimal solutions for small or medium instances. However, when the number of tasks, products, routes or resources increases, exact methods often become computationally expensive.

For this reason, many authors propose heuristic or metaheuristic methods. Zhang et al. [54] used an improved artificial bee colony algorithm for integrated remanufacturing process planning and scheduling. Zhang et al. [55] relied on an improved spider monkey optimization approach for an energy-aware formulation. Zhang et al. [56] proposed an improved grey wolf optimization method for flexible job-shop remanufacturing scheduling. Tian et al. [43] considered an energy-efficient remanufacturing scheduling problem with a lot-streaming production mode. These methods do not always guarantee optimality, but they are able to provide good-quality solutions within acceptable computational time.

Uncertainty and dynamic decision-making have also been progressively introduced in remanufacturing scheduling. Fu et al. studied a stochastic integrated disassembly–reprocessing–reassembly scheduling problem where processing times are modeled through probability distributions [15]. Yuan et al. considered fuzzy processing times in capacitated disassembly scheduling, which is useful when exact probability distributions are not available [52]. Han et al. proposed a deep reinforcement learning approach for dynamic disassembly scheduling, showing that learning-based methods can support decisions when product quantities, buffers or system states evolve over time [22].

These contributions show that heuristic, metaheuristic and learning-based approaches are well suited for complex disassembly and remanufacturing problems. This point is directly related to the methodology adopted in the present work. Exact mathematical models provide a rigorous formulation of the multi-product human–robot collaborative disassembly line. However, because the problem combines task assignment, station opening, mode selection, precedence relations and resource constraints, exact resolution becomes increasingly difficult as the number of products and tasks grows. This limitation supports the need for a scalable metaheuristic solution method after the exact modelling phase.

2.3 Electric Vehicle Battery Disassembly: Specific Challenges and Research Trends

After introducing disassembly as a general decision-making problem in remanufacturing systems, it is important to focus on the particular case of electric vehicle batteries. The disassembly of end-of-life electric vehicle batteries cannot be considered as a classical product disassembly problem. Battery packs are complex electrochemical systems that include cells, modules, busbars, cooling systems, electronic control units, safety devices, covers, connectors and several types of mechanical fasteners. Their structure depends strongly on the manufacturer, the vehicle model and the battery generation. Therefore, the disassembly process must deal with a high level of product heterogeneity.

In addition to structural complexity, electric vehicle batteries raise important safety issues. Even at the end of their first life, they may still contain residual energy and present electrical, thermal and chemical risks. Operations such as removing high-voltage con-

nectors, separating modules, handling damaged cells or accessing internal components must be performed carefully in order to avoid short circuits, thermal runaway or operator exposure to hazardous materials. Harper et al. emphasized that the recycling of lithium-ion batteries from electric vehicles requires careful upstream handling and disassembly before material recovery, because the design and composition of battery packs have a direct impact on the safety and efficiency of recycling routes [23]. Similarly, Mossali et al. highlighted that battery recycling opportunities depend not only on downstream metallurgical processes, but also on the ability to prepare and separate battery components efficiently before recycling [34].

2.3.1 Specificity of End-of-Life EV Battery Disassembly

The specificity of electric vehicle battery disassembly is mainly related to three aspects: product architecture, safety constraints, and end-of-life uncertainty. First, battery packs are not standardized. Different manufacturers use different cell formats, module arrangements, cooling systems, enclosure designs, connection technologies and fastening methods. Kampker et al. proposed a framework for classifying battery system architectures, showing that battery packs can vary considerably in their structural organization and integration level [26]. Belingardi and Scattina also discussed the integration of battery packs into the vehicle structure, which confirms that battery accessibility and disassembly depend on architectural and design choices made at the vehicle level [6].

Second, battery disassembly includes hazardous tasks that cannot be assigned only according to processing time or cost. Some operations require specific safety procedures and may be more suitable for robotic execution, while others require human flexibility and judgment. For example, removing high-voltage components, handling damaged modules, or operating near potentially unstable cells may expose workers to serious risks. This explains why recent studies increasingly consider the allocation of hazardous tasks to robots and complex tasks to humans, instead of relying only on manual disassembly.

Third, end-of-life batteries are not always received in the same condition. Their state may depend on their usage history, state of health, accident history, level of degradation, corrosion, deformation or previous maintenance operations. Consequently, the time needed to perform the same disassembly task may vary from one battery to another. This point is particularly important for the present work, since a deterministic model with fixed operation times can be useful as a first modeling step, but it does not fully represent the variability of real disassembly operations.

2.3.2 Manual, Automated, and Human–Robot Assisted Disassembly

Current industrial battery disassembly is still largely manual. Manual disassembly remains attractive because humans are flexible and able to adapt to product variability, unexpected defects and incomplete product information. However, manual operations expose workers to ergonomic and safety risks, especially when handling heavy modules, sharp components, damaged batteries or high-voltage elements. Moreover, manual disassembly can be time-consuming and difficult to standardize.

For these reasons, several authors have investigated automated and robotised disassembly solutions. Villagrossi and Dinon studied the use of robotics for electric vehicle battery pack disassembly in the context of sustainable remanufacturing. They showed that robotic systems can support operations that are repetitive, physically demanding or dangerous, but also underlined that full automation remains difficult because of prod-

uct variability, uncertainty and the complexity of battery structures [44]. Kaarlela et al. also reviewed robotised disassembly of electric vehicle batteries and concluded that automation requires flexible perception, adaptable tooling, safe manipulation strategies and better product information [25].

Between fully manual and fully automated disassembly, human–robot collaboration appears as an intermediate and promising solution. In such systems, humans and robots do not replace each other completely. Instead, each resource is used according to its strengths. Human operators can handle tasks requiring perception, reasoning and adaptation, while robots can perform dangerous, repetitive, heavy or highly controlled operations. Rettenmeier et al. described disassembly technologies for end-of-life automotive battery packs as a key element of the circular battery value chain, and emphasized that future disassembly systems must combine safety, flexibility and process efficiency [40]. This supports the idea that collaborative or hybrid configurations may be more realistic than fully automated systems in the short and medium term.

2.3.3 Main Research Directions in EV Battery Disassembly

The literature on electric vehicle battery disassembly can be organized around several research directions. A first group of studies focuses on the physical and technological aspects of disassembly. These works analyze battery structures, disassembly procedures, tools, fasteners, accessibility and safety requirements. Wu et al. [46] reviewed lithium-ion battery disassembly for recycling and emphasized the importance of safe and efficient upstream dismantling. Rettenmeier et al. [40] analyzed battery pack disassembly technologies from a process-oriented perspective. Kaarlela et al. [25] focused on robotised disassembly and the requirements for flexible automation. Neri et al. [35] also reviewed sustainable management of EV battery remanufacturing, showing that disassembly, diagnostics and recovery-route selection are closely connected.

A second group focuses on disassembly planning and optimization. These studies formulate the disassembly process as a sequence planning, scheduling or line balancing problem. The objective is to determine a feasible and efficient organization of tasks while respecting precedence relations. For instance, Baazouzi et al. optimized disassembly strategies for electric vehicle batteries by considering not only the sequence of component removal, but also the depth of disassembly and the circular economy strategy associated with each component [3]. This type of work shows that the best disassembly strategy does not only depend on technical feasibility, but also on economic value and recovery decisions.

A third group of studies investigates artificial intelligence and data-driven methods for battery disassembly. Ai et al. [1] reviewed the use of artificial intelligence in electric vehicle battery disassembly. Li et al. [29] reviewed AI-driven end-of-life electric vehicle battery disassembly and discussed applications such as state-of-health assessment, sequence planning, robotic perception, decision-making and human–robot collaboration. These approaches are promising because they can help deal with product heterogeneity and uncertain end-of-life conditions. However, they also require reliable data, detailed product information and robust models that can generalize across different battery designs.

Finally, a more recent stream of research focuses on human–robot collaborative disassembly line balancing. In this direction, the problem is no longer limited to finding a disassembly sequence. It also includes the assignment of tasks to humans, robots or collaborative modes, the opening or use of workstations, and the satisfaction of cycle time and

precedence constraints. This evolution is directly related to the present research, where the disassembly of several types of batteries is modeled in a human–robot collaborative line. The deterministic version of the model provides a first structured representation of the system, while the stochastic extension is introduced later to account for the variability of human operation times.

2.4 Disassembly Sequence Planning for EV Batteries

Disassembly sequence planning is one of the most studied problems in the field of end-of-life electric vehicle battery recovery. Its purpose is to determine a feasible and efficient order for removing components while respecting technological and physical precedence constraints. In the case of battery packs and modules, this problem is particularly important because a wrong sequence may increase the disassembly time, reduce the value of recovered components, create unnecessary tool changes, or even increase safety risks. Therefore, before dealing with line balancing or resource allocation, many studies first focus on the construction of a valid and optimized disassembly sequence.

The disassembly sequence planning problem is generally modeled using graphs or matrices. Components or operations are represented as nodes, while precedence relations, contact relations or accessibility constraints are represented as arcs or matrix values. Once this representation is built, the optimization method searches for one or several feasible sequences according to a given objective. Depending on the study, the objective may be to reduce disassembly time, maximize recovery profit, reduce tool changes, minimize energy consumption, or remove hazardous parts as early as possible.

2.4.1 Sequence, Depth, and Circular Strategy Optimization

A relevant contribution in this direction was proposed by Baazouzi et al., who studied the optimization of disassembly strategies for electric vehicle batteries. Their work is different from classical sequence planning because it does not only determine the order of disassembly operations. It also integrates the depth of disassembly and the circular economy strategy associated with each component. In other words, the model decides whether the battery should be fully disassembled or whether the process should stop at a certain level, and it also determines whether each recovered component should be reused, remanufactured, repurposed or recycled [3].

This contribution is important because complete disassembly is not always the best option. In some cases, continuing the disassembly process may require more time and cost than the value recovered from the additional components. The optimal strategy therefore depends on the condition of the battery and the economic value of its components. This is particularly relevant for end-of-life batteries, where the state of modules, covers, cooling systems or electronic parts may vary significantly from one unit to another.

Techno-economic and environmental disassembly planning also follows this logic. Alfaro-Algaba and Ramirez studied the disassembly planning of lithium-ion electric vehicle battery packs for remanufacturing by considering both profitability and environmental impact [2]. This type of contribution shows that the decision to continue or stop disassembly should be linked to the value of recovered components, the cost of operations, and environmental consequences. However, these approaches generally remain focused on planning or strategy selection rather than on the detailed balancing of a multi-product collaborative line.

2.4.2 Graph-Based and Matrix-Based Precedence Representation

Most disassembly sequence planning studies rely on a structured representation of the product. This representation is necessary because battery components cannot be removed in any arbitrary order. For example, covers, screws, busbars, sensors or cooling elements may block access to internal modules. As a result, precedence constraints are required to describe which operations must be performed before others.

Ke et al. proposed a disassembly sequence planning method for electric vehicle batteries based on a frame-subgroup structure combined with a genetic algorithm. The authors first construct an improved disassembly relation hybrid graph and a disassembly relation matrix to represent both precedence and connection relationships between components. Then, they identify a particular structure called frame-subgroup, where several components are connected to a central frame but are relatively independent from each other. This structure makes it possible to reduce the search space, since several components can be permuted without violating the disassembly logic [27].

The main advantage of this approach is that it simplifies the optimization process. By using Cantor expansion, the authors transform the permutation-based sequence problem into a positive integer representation. This allows the genetic algorithm to manipulate solutions more easily, with simpler encoding and decoding procedures [27]. However, the effectiveness of this method depends on the presence of a frame-subgroup structure in the product. Therefore, it is particularly useful for certain battery architectures, but it may be less general when the product structure does not contain this type of organization.

Other works also use graph-based representations to handle precedence, accessibility and tool-related constraints. Zhan et al. developed an environment-oriented disassembly planning approach for end-of-life vehicle batteries based on a hybrid graph and an improved Northern Goshawk Optimization algorithm [53]. Meng et al. proposed a priority-constraint-graph-based method for lithium battery pack disassembly sequence optimization, considering criteria such as turns, tool changes and sequence rationality [32]. These studies confirm that graph and matrix representations remain central in battery disassembly planning.

Graph-based and matrix-based representations are also useful because they provide a direct link between the physical structure of the battery and the mathematical optimization model. This idea is important for the present research, since the proposed model also relies on precedence information to guarantee feasible task assignment and sequencing. However, unlike pure sequence planning studies, the present work does not stop at finding an order of component removal. It also considers stations, human and robotic resources, execution modes, and line capacity constraints.

2.4.3 Risk-Aware and Environment-Oriented Sequence Planning

Because battery disassembly involves high-voltage components and potentially hazardous materials, several authors have introduced risk and environmental criteria into sequence planning. Zhan et al. minimized the hazard index and energy consumption for end-of-life vehicle batteries, showing that the best sequence is not necessarily the fastest one, but the one that better balances safety and energy performance [53]. Yang et al. proposed an integrated risk-aware smart disassembly planning approach for scrap electric vehicle batteries, considering recovery profit, energy consumption and risk in the same planning framework [50].

These contributions are important because they show that battery disassembly cannot be evaluated only by feasibility or processing time. A sequence that removes risky

components early may be preferable even if it is not the shortest sequence. Similarly, reducing tool changes or unnecessary manipulations can improve operational safety and reduce energy use. Nevertheless, these works generally remain focused on sequence planning for one battery type, while the present research focuses on line balancing and resource allocation for several battery types.

2.4.4 Knowledge-Based and AI-Based Sequence Planning

More recent studies have started to use artificial intelligence and knowledge-based representations to improve disassembly sequence planning. This evolution is motivated by the fact that battery disassembly involves heterogeneous information: components, tools, connection types, risks, operation times, accessibility relations and possible recovery options. Classical precedence matrices can represent the order constraints, but they may not capture all the semantic relations between battery elements.

Yin et al. proposed a sequential disassembly planning approach based on knowledge graph and graph isomorphism network for supporting power battery remanufacturing. Their framework combines TransH knowledge graph embedding, Graph Isomorphism Network and MOEA/D to learn complex relations between battery components, tools and disassembly operations [51]. The use of a knowledge graph allows the model to represent richer information than a simple precedence matrix, while the graph neural network helps to learn structural dependencies in the disassembly process.

This type of approach is promising because it can support more adaptive and information-rich disassembly planning. It may be especially useful when the same system must deal with different battery structures or when product information needs to be updated. However, such methods often require detailed data and a more complex computational framework. For this reason, they are still difficult to apply directly in a simple mathematical model intended to define a first optimization framework.

2.4.5 Limits of Sequence Planning Approaches

Although disassembly sequence planning is an essential research direction, it does not cover the whole decision problem faced in a disassembly line. A sequence gives the order in which operations should be performed, but it does not necessarily specify how tasks are assigned to stations, whether a task is performed by a human, a robot or both, or how the workload is balanced across the line. In other words, sequence planning is necessary, but not sufficient when the objective is to design or operate a complete disassembly line.

Another limitation is that many sequence planning studies focus on a single battery type. The case study usually concerns one specific battery pack or module, such as an Audi, Tesla, Volkswagen or another battery architecture. This allows a detailed analysis of the product, but it does not fully represent the situation of industrial treatment centers, where several battery types may arrive and need to be processed using the same resources.

A further limitation concerns the deterministic nature of many models. Operation times, costs, energy consumption and risks are often assumed to be known in advance. This assumption simplifies the optimization problem, but it does not fully reflect end-of-life battery disassembly, where the same operation may take different times depending on the condition of the battery, the accessibility of fasteners, corrosion, deformation or operator performance. This is one of the reasons why the present research first develops a deterministic model as a structured baseline, and then extends it into a stochastic version where uncertainty is introduced through human operation times.

2.5 Disassembly Line Balancing for EV Batteries

Disassembly sequence planning provides the order in which the components of a product should be removed. However, when the objective is to organize an industrial disassembly system, the sequence alone is not sufficient. The decision-maker must also determine how the tasks are distributed among workstations, which resources are assigned to each workstation, and how the workload is balanced along the line. This leads to the disassembly line balancing problem, which is a natural extension of classical assembly line balancing to end-of-life products.

In a disassembly line balancing problem, a set of disassembly tasks must be assigned to workstations while respecting precedence constraints and a given cycle time. The aim is usually to obtain an efficient line configuration by reducing idle times, limiting the number of workstations, improving workload balance, reducing cost, or increasing safety. In the context of electric vehicle batteries, this problem becomes more complex because some tasks are hazardous, some require human flexibility, and others may be suitable for robotic execution. Therefore, task assignment cannot be based only on processing time; it must also consider the nature of the operation and the compatibility between tasks and resources.

2.5.1 From Sequence Planning to Line Balancing

The transition from disassembly sequence planning to disassembly line balancing represents an important step toward practical implementation. While sequence planning determines a feasible order of operations, line balancing transforms this order into a production organization. It defines which tasks are performed at each station and ensures that the total workload assigned to a station does not exceed the cycle time. This is particularly important when the disassembly process is repeated for several batteries and must be organized as a line rather than as a single isolated operation.

For electric vehicle batteries, line balancing has been studied to improve the efficiency and safety of module or pack disassembly. Wu et al. proposed a cell-level disassembly line balancing model for waste power battery modules in a human-machine hybrid mode. In their work, tasks are classified according to their characteristics: hazardous tasks are assigned to robots, complex tasks are assigned to humans, and ordinary tasks can be assigned to either humans or robots [48]. This classification is important because it introduces a practical link between the physical nature of the disassembly task and the resource that should perform it.

Their model considers a line composed of manual and robotic workstations, and the balancing problem is solved using NSGA-II. Although the study focuses on a specific battery module, it shows that battery disassembly line balancing cannot be treated as a classical line balancing problem. The presence of hazards, task complexity and resource compatibility must be explicitly integrated into the decision process [48]. This idea is also relevant for the present work, where each task can be performed in a specific execution mode: human, robot, or human-robot collaboration.

2.5.2 Manual, Robotic, Hybrid and Collaborative Workstations

A key issue in battery disassembly line balancing is the choice of workstation type. In a purely manual line, all tasks are performed by human operators. This configuration offers flexibility, but it exposes workers to safety risks and may lead to high variability

in operation times. In a robotic line, tasks are performed by robots, which can improve safety for dangerous operations but may have difficulty dealing with product variability and unexpected conditions. Hybrid or collaborative configurations aim to combine the advantages of both resources.

Wu et al. introduced a human-machine hybrid mode where human and robotic workstations coexist in the same disassembly line [48]. This approach is useful because it avoids assigning all operations to one type of resource. Instead, each task is directed toward the most suitable resource according to its difficulty and hazard level. However, in this type of hybrid system, a task is generally assigned either to a human or to a robot, and the collaboration between the two resources is not always modeled as a simultaneous execution mode.

Human-robot collaborative disassembly goes further by considering that some operations may require the simultaneous intervention of a human and a robot. Chu and Chen proposed a human-robot collaboration disassembly planning model for end-of-life power batteries [12]. Zhang et al. also studied a multi-objective human-robot collaboration disassembly planning approach for retired power batteries based on entropy-weight fuzzy hierarchical analysis and a differential evolution algorithm [58]. These contributions show that human-robot cooperation can improve the feasibility and efficiency of battery disassembly when tasks are properly assigned.

More recently, Qin et al. studied a multiple product hybrid disassembly line balancing problem with human-robot collaboration [39]. This work is particularly relevant because it introduces the multi-product dimension into a human-robot collaborative disassembly line. The authors consider several product types and integrate task assignment, station assignment and resource decisions in the same framework. Their contribution confirms that the multi-product dimension is becoming increasingly important in disassembly line balancing, since real recovery facilities rarely process only one product type.

2.5.3 Line Configuration and Operational Performance

Some studies focus on the physical configuration of the dismantling line. Mao et al. optimized dismantling lines for used electric vehicle battery packs by comparing unilateral and symmetric bilateral line configurations [31]. Their model considers the number of workstations, idle time, risk and disassembly cost. This contribution is useful because it shows that the line layout itself can affect operational performance. However, the present research does not focus on comparing physical layouts. It focuses on the assignment of tasks, stations, resources and execution modes in a multi-product collaborative line.

Other line balancing works introduce remanufacturing demand, hazard characteristics, destructive operations or environmental constraints. Zhu et al. studied a multi-objective human-robot collaborative disassembly line balancing problem considering component remanufacturing demand and hazard characteristics [61]. These contributions confirm that realistic disassembly line balancing increasingly considers more than simple workload distribution. Nevertheless, adding many objectives or operational features can make the model more difficult to interpret and solve.

2.5.4 Cost-Oriented and Performance-Oriented Line Balancing Criteria

The objectives considered in disassembly line balancing vary from one study to another. Some authors focus on minimizing the number of workstations or reducing idle time in order to improve line efficiency. Others include cost, risk, energy consumption, profit

or environmental indicators. In battery disassembly, safety and cost are particularly important because the process involves hazardous operations and expensive resources.

In the model proposed by Wu et al. [48], several criteria are considered, including the number of workstations, workload balance, the number of human workers and disassembly cost. Zhan et al. [53] considered hazard and energy indicators in environment-oriented sequence planning. Mao et al. [31] included workstation number, idle time, risk and disassembly cost when comparing dismantling line configurations. Zhu et al. [60] considered stochastic and sequence-dependent effects in disassembly line balancing. This reflects the fact that a good disassembly line must be efficient, safe and economically acceptable at the same time. However, including many objectives may make the model more difficult to interpret and solve. In industrial applications, it may also be useful to aggregate the main economic components into a single objective function, especially when the purpose is to obtain a clear and operational decision.

The present work follows this cost-oriented perspective. Instead of formulating the problem as a multi-objective model, it uses a mono-objective economic formulation. The objective is to minimize a total cost that includes station opening costs and task execution costs. In the stochastic extension, an expected recourse cost is also added to penalize possible overloads of workstations under uncertain human operation times. This choice makes it possible to keep a clear optimization criterion while still representing important operational decisions.

2.6 Human–Robot Collaboration in Battery Disassembly

The disassembly of end-of-life electric vehicle batteries requires both flexibility and safety. Human operators are able to adapt to product variability, unexpected damage, missing information and complex manual operations. Robots, on the other hand, are more suitable for repetitive, heavy, precise or hazardous tasks. For this reason, human–robot collaboration has become an important research direction in battery disassembly. It offers an intermediate solution between fully manual disassembly, which exposes workers to risks, and full automation, which remains difficult because of the diversity and uncertainty of end-of-life battery packs.

Several studies emphasize that full automation of battery disassembly is still challenging. Battery packs differ in architecture, fastening methods, cooling systems, module arrangement and accessibility. In addition, the condition of end-of-life batteries may be uncertain, especially when batteries have been damaged, aged or exposed to harsh operating conditions. Villagrossi and Dinon showed that robotic systems are promising for sustainable battery remanufacturing, but they also pointed out that robotized disassembly must deal with product variability, perception issues and complex manipulation tasks [44]. Kaarlela et al. reached a similar conclusion in their review of robotised electric vehicle battery disassembly, noting that flexible automation requires adaptable tooling, safe control, and reliable information about the product structure [25].

Human–robot collaboration is therefore not only a technological choice, but also a modeling issue. The disassembly system must decide which tasks are performed by humans, which tasks are performed by robots, and which tasks require cooperation between the two resources. These decisions affect safety, workload balance, station configuration, cycle time feasibility and economic performance. In this context, the mathematical model must represent not only the precedence relations between tasks, but also the compatibility between each task and each execution mode.

2.6.1 From Human–Machine Hybrid Systems to Real Collaboration

In the literature, it is important to distinguish between human–machine hybrid systems and real human–robot collaboration. In a hybrid system, humans and robots may coexist in the same line, but each task is generally assigned to either a human or a robot. In this case, the two resources are used separately. In a collaborative system, some tasks can be performed jointly by a human and a robot, which means that both resources are required at the same time for the same operation.

Wu et al. proposed a human–machine hybrid disassembly line balancing model for waste power battery modules [48]. In their model, hazardous tasks are assigned to robots, complex tasks are assigned to humans, and ordinary tasks can be assigned to either type of workstation. This classification is useful because it reflects the practical differences between task categories. However, the model mainly represents a hybrid allocation logic, where tasks are distributed between human and robotic stations, rather than a fully collaborative execution mode.

Chu and Chen went further by studying human–robot collaboration disassembly planning for end-of-life power batteries [12]. Their work explicitly considers collaborative disassembly and shows how task allocation and sequencing can be organized when humans and robots are both involved in the process. Gao et al. proposed a partially observable deep reinforcement learning approach for multi-agent strategy optimization in human–robot collaborative battery disassembly [16]. These works show that HRC battery disassembly can be approached either through mathematical optimization or through learning-based decision-making.

2.6.2 Task Classification in HRC Disassembly

A central element in HRC disassembly models is the classification of tasks. In battery disassembly, not all operations have the same level of risk, complexity or automation potential. Some tasks are hazardous because they involve high-voltage components, unstable cells or dangerous materials. These tasks are often more suitable for robotic execution in order to reduce operator exposure. Other tasks are complex because they require perception, dexterity, adaptation or decision-making. These tasks may remain more appropriate for human operators. Finally, some tasks are ordinary or flexible and can be performed either by a human or by a robot, depending on the line configuration.

The classification proposed by Wu et al. is based on this logic, with hazardous tasks, complex tasks and ordinary tasks [48]. This classification makes the task assignment decision more realistic than a simple time-based allocation. It also shows that safety and task nature must be integrated into disassembly planning. In the same direction, Chu and Chen considered task categories compatible with human execution, robot execution or collaboration [12]. These categories help define the feasible execution modes of each task.

The present research follows this idea by classifying battery disassembly tasks into four groups: human-only tasks, robot-only tasks, collaborative human–robot tasks, and normal tasks that can be performed either by a human or by a robot. This classification makes it possible to represent both technological restrictions and operational flexibility. Human-only tasks correspond to operations that require human dexterity or judgment. Robot-only tasks correspond to operations that are dangerous or better suited to automated execution. Collaborative tasks require both resources simultaneously. Normal tasks remain flexible and can be assigned to the most suitable resource according to the optimization result.

This classification is particularly important in a mono-objective cost-oriented model. Since the model minimizes a global economic cost, it must still respect the feasible execution modes of each task. In other words, the cost minimization cannot assign a hazardous task to a human-only mode if the task is defined as robot-only, and it cannot assign a collaborative task to a station that does not contain both a human and a robot. Therefore, task classification is not only descriptive; it directly shapes the feasible solution space of the optimization model.

2.6.3 Synchronization and Resource Constraints in Collaborative Stations

When a task is performed in collaboration, the model must ensure that the human and the robot are available at the same station and during the same time interval. This creates a synchronization requirement that does not exist in purely manual or purely robotic tasks. A collaborative operation cannot start if only one of the two resources is available. It also blocks both resources during its execution. Therefore, collaborative tasks increase the complexity of the scheduling and line balancing problem.

In HRC disassembly planning, this synchronization is usually handled through resource constraints and non-overlapping constraints. If two tasks use the same human resource in the same station, they cannot overlap. The same rule applies to robotic resources. For collaborative tasks, both sets of constraints are activated at the same time, because the task uses both the human and the robot. Chu and Chen highlighted the importance of such constraints in human–robot collaborative battery disassembly planning [12]. Qin et al. also considered human–robot collaboration in a multiple product disassembly line balancing framework, where the availability of resources and the assignment of tasks to workstations are jointly considered [39].

These modeling aspects are directly related to the present work. The proposed deterministic model assigns each task to a station and to one execution mode. If the task is collaborative, the station must contain both a human and a robot. In addition, tasks sharing the same human or robotic resource in the same station cannot overlap. This ensures that the planning remains physically feasible. In the stochastic extension, the same first-stage assignment decisions are kept, but the execution times of human tasks vary according to scenarios. This makes the synchronization and resource feasibility even more important, because the human workload may change from one scenario to another.

2.6.4 Learning-Based HRC and Remaining Implementation Issues

Recent studies also explore reinforcement learning and multi-agent learning for disassembly systems. Guo et al. reviewed reinforcement learning applications for disassembly system optimization, including sequence planning, line balancing and human–robot collaboration [20]. Gao et al. modeled HRC battery disassembly as a partially observable multi-agent decision problem and used deep reinforcement learning to optimize collaborative strategies [16]. These works are promising because they can handle dynamic decisions and incomplete information.

However, learning-based methods often require a simulation environment, large amounts of training data and careful definition of states, actions and rewards. This can make them difficult to use as a first modeling framework when the goal is to clearly formulate the structure of the optimization problem. For this reason, the present research focuses first on mathematical modeling. The deterministic model defines the structural

decisions, and the stochastic extension introduces uncertainty in human operation times. A metaheuristic solution method can then be developed to solve larger instances without requiring a full learning environment.

2.7 Uncertainty in Disassembly Planning

Most disassembly planning and line balancing models are first formulated in a deterministic way. In these models, operation times, costs, resource requirements and precedence relations are assumed to be known in advance. This assumption is useful because it makes the problem easier to formulate and solve. It also provides a clear baseline for understanding the structure of the disassembly system. However, in real end-of-life product recovery, and especially in electric vehicle battery disassembly, several parameters are uncertain.

The deterministic assumption becomes particularly restrictive when dealing with used batteries. Two batteries of the same model may not require exactly the same disassembly time, because their condition depends on their usage history, degradation level, accident history, corrosion, deformation, residual charge, or the state of fasteners and connectors. Therefore, a deterministic model can provide an initial feasible organization of the line, but it may not fully represent the variability observed in practice. This is why uncertainty modeling has become an important research direction in disassembly planning.

2.7.1 Sources of Uncertainty in End-of-Life Battery Disassembly

The main source of uncertainty in battery disassembly is the condition of the returned product. Unlike new products in assembly systems, end-of-life batteries arrive after several years of use and may have experienced different operating conditions. Some batteries may be intact, while others may be damaged, corroded, deformed or partially degraded. These differences directly affect the difficulty of disassembly operations.

Operation times are therefore among the most uncertain parameters. For example, removing a cover, disconnecting a cable, unscrewing fasteners or extracting a module may take more time than expected if the component is blocked, damaged or difficult to access. Human operation times are especially variable because they depend not only on the product condition, but also on the operator's experience, fatigue, judgment and adaptation to unexpected situations. This variability is one of the reasons why manual disassembly remains flexible but difficult to standardize.

Uncertainty may also affect safety and resource assignment. A task that is normally simple may become risky if the battery is damaged or if a component is not accessible in the expected way. Similarly, a task initially planned for human execution may require robotic assistance if the risk level increases. Villagrossi and Dinon [44] pointed out that product variability remains a major barrier for robotized battery pack disassembly. Kaarlela et al. [25] also emphasized the need for adaptable tooling and reliable product information in robotised battery disassembly. Li et al. [29] highlighted that AI-driven disassembly still depends on reliable data and robust decision-making methods. These observations justify the need for models that can move beyond fixed deterministic parameters.

2.7.2 Fuzzy, Stochastic, and Scenario-Based Approaches

Several modeling approaches have been proposed in the literature to represent uncertainty in disassembly and remanufacturing systems. A first approach consists of using fuzzy numbers. In this case, operation times are not represented by a single fixed value, but by an interval or a triangular fuzzy number. This allows the model to describe uncertain task durations when exact probability distributions are not available. Yuan et al. used fuzzy processing times in a capacitated multi-objective disassembly scheduling problem, showing that fuzzy modeling can be useful when disassembly durations are imprecise rather than exactly random [52].

A second approach is stochastic modeling. Here, uncertain parameters are represented by probability distributions. Operation times may follow normal, uniform or other distributions depending on the available data and assumptions. Fu et al. studied stochastic multi-objective integrated disassembly–reprocessing–reassembly scheduling with processing times following normal distributions [15]. This type of work is important because it shows that uncertainty should be considered not only in isolated disassembly tasks, but also in integrated remanufacturing systems.

A third approach is scenario-based optimization. Instead of solving the problem for one deterministic value of each parameter, several scenarios are generated. Each scenario represents one possible realization of uncertain operation times. The optimization model then searches for a solution that performs well over all scenarios, or in expectation. This approach is particularly useful when uncertainty affects the processing times but the main structural decisions, such as task assignment and station opening, must be made before the real condition of the product is fully known.

This scenario-based logic is consistent with the present research. The deterministic model is first used to define the basic structure of the multi-product human–robot collaborative disassembly line. Then, the stochastic extension introduces uncertainty in human operation times. The first-stage decisions, such as station opening, task assignment, execution mode and resource allocation, remain common to all scenarios. The second-stage variables, such as task starting times, finishing times and recourse variables, are scenario-dependent.

2.7.3 Stochastic Disassembly Line Balancing Models

In the context of human–robot collaborative disassembly, Guo et al. studied a disassembly line balancing problem with stochastic operation times and proposed a solution based on a multi-objective shuffled frog leaping algorithm [21]. Their work is important because it directly recognizes that operation times in HRC disassembly are not always deterministic. It also shows that stochastic task durations can significantly influence the balancing of the line and the quality of the obtained solutions.

Other stochastic disassembly line balancing studies consider uncertain task times, energy consumption or sequence-dependent effects. Zhu et al. proposed a stochastic sequence-dependent disassembly line balancing problem and solved it using a multi-objective adaptive large neighbourhood search algorithm [60]. Their work accounts for uncertain operation times and sequence-dependent effects, which makes the planning problem closer to real end-of-life conditions. Zhang et al. studied an energy-oriented stochastic disassembly line balancing problem where task times and energy consumption are random and the objective is to reduce total energy consumption [57]. Wu et al. proposed a hybrid evolutionary algorithm for a stochastic HRC disassembly line balancing problem considering carbon emission optimization [47]. These works support the idea

that deterministic processing times are often insufficient for realistic disassembly line design.

The stochastic extension of the present work follows this idea, but with a mono-objective economic formulation. Instead of treating the stochastic problem as a multi-objective optimization problem, the model minimizes a total expected cost. This cost includes station opening costs, task execution costs and an expected recourse cost associated with possible station overloads in the scenarios. In this way, the uncertainty is not ignored, but it is translated into an economic penalty.

The recourse mechanism is an important modeling choice. In the deterministic model, each opened station must respect the cycle time as a strict capacity constraint. In the stochastic version, the station load may exceed the cycle time in some scenarios, but only within a bounded margin and with a penalty cost. This makes the model less rigid and closer to real operational conditions. Instead of forcing the line to be feasible for the worst possible realization of all human times, the model allows limited deviations and penalizes them economically.

2.7.4 Limits of Existing Stochastic Approaches

Although uncertainty has been increasingly considered in disassembly planning, several limitations remain. First, many stochastic or fuzzy models are still developed for general products rather than for electric vehicle batteries. As a result, they do not always include battery-specific constraints such as hazardous operations, high-voltage components, or human-robot collaboration.

Second, stochastic models often focus on one product type. This limits their applicability in real recovery centers, where several battery types may be processed on the same line. Each battery type has its own tasks, precedence relations, durations and resource requirements. Therefore, combining uncertainty with a multi-product structure remains a relevant research direction.

Third, many existing studies use uncertainty mainly to evaluate or compare solutions, while the integration of uncertainty into the economic structure of the model is less frequent. In practical terms, uncertainty should not only be treated as a technical variation, but also as a cost-related issue. Delays, overloads, additional labor time or operational disruptions can all generate economic consequences.

For these reasons, the present work adopts a progressive modeling strategy. A deterministic mono-objective model is first developed in order to define the basic assignment, station opening and resource allocation decisions. Then, a stochastic scenario-based model is introduced to better represent the variability of human operation times. This extension keeps the same structural decisions across scenarios, while allowing scenario-dependent scheduling and bounded recourse on station loads. This approach makes it possible to preserve the clarity of the deterministic model while improving its realism.

2.8 Multi-Product Disassembly Lines

Most studies on electric vehicle battery disassembly focus on a single battery pack or a single battery module. This choice is understandable because each battery type has its own structure, components, fastening methods and precedence relations. Working on one product makes it possible to describe the disassembly process in detail and to build a specific graph or matrix of precedence constraints. However, in real recovery facilities,

end-of-life batteries do not necessarily belong to the same model. A dismantling center may receive several battery types with different architectures, operation times and safety requirements. This industrial reality motivates the study of multi-product disassembly lines.

A multi-product disassembly line is more complex than a single-product line because the model must handle several task sets and several precedence structures at the same time. Each product has its own disassembly graph, but the products may share the same stations, operators, robots and cycle time. Therefore, the optimization problem must determine how the tasks of different products are assigned to the available workstations while preserving the technological logic of each product. This makes the problem closer to a mixed or multi-model production environment, but with additional difficulties related to end-of-life uncertainty, product condition and safety constraints.

2.8.1 Motivation for Multi-Product Disassembly

The need for multi-product modeling is particularly clear in the case of electric vehicle batteries. Battery packs are not standardized across manufacturers. Their internal structure depends on the cell format, the module arrangement, the cooling system, the type of enclosure, the fastening technologies and the electrical connections. Kampker et al. showed that battery system architectures can be classified according to several structural and functional characteristics, which confirms the high diversity of battery pack designs [26]. This diversity directly affects disassembly planning because a task sequence defined for one battery type cannot be applied automatically to another one.

From an operational point of view, treating each battery type with a completely separate line would not always be economically feasible. It would require several dedicated infrastructures, each designed for a specific battery model. A more flexible approach consists of using the same line to process different battery types, while adapting the task assignment and execution modes according to each product. This is why multi-product disassembly planning is important for future battery recovery systems.

Multi-product disassembly also creates an important modeling challenge. If each battery type has a different number of tasks and different precedence relations, the model must keep the identity of each product. It is not possible to merge all tasks into a single generic sequence without losing the technological structure of each battery. Therefore, the product index becomes essential in the mathematical formulation. This idea is also adopted in the present research, where each battery type has its own task set, precedence matrix and operation times.

2.8.2 Multi-Product Modeling Approaches

In the broader disassembly literature, Ehm proposed a scheduling and process planning model for dismantling systems with flexible disassembly modes and recovery levels [14]. This type of work is useful because it shows how disassembly decisions can be formulated for systems where different products or recovery requirements must be considered. The model integrates the selection of disassembly operations, the choice of recovery level and the scheduling of tasks, which makes it relevant for studying more realistic end-of-life product recovery systems.

For electric vehicle batteries, Qin et al. studied a multiple product hybrid disassembly line balancing problem with human-robot collaboration [39]. Their work is one of the closest contributions to the present research because it explicitly combines a multi-product

setting with human–robot resources. In such a framework, the line must process several products while deciding how tasks are distributed among stations and resources. This confirms that the combination of multi-product disassembly and HRC is a recent and relevant research direction.

A common way to represent several products is to keep a separate precedence structure for each product. Each battery type is described by its own precedence matrix or graph. Then, the optimization model uses a product index to distinguish the tasks and constraints associated with each battery type. This approach avoids creating artificial precedence relations between tasks belonging to different products. In the present work, this modeling logic is adopted by defining a task set for each product and by formulating precedence constraints separately for each battery type.

The use of product proportions or product weights is also important in multi-product disassembly. In a real treatment center, all battery types may not arrive with the same frequency. Some types may represent a larger share of the incoming flow, while others may be less frequent. For this reason, a multi-product model can use product weights to represent the product mix. This allows the objective function to reflect the relative importance of each battery type in the overall system. In the present deterministic model, the execution cost of tasks is weighted by the proportion of each product type in the product mix.

2.8.3 Destructive Operations, Parallel Layouts and Industrial Extensions

Recent multi-product studies extend the classical disassembly line balancing problem by adding destructive operations, preventive maintenance, parallel layouts or environmental criteria. Zhang et al. studied reliability and efficiency optimization of multi-product disassembly lines with destructive operations and preventive maintenance [59]. Guo et al. proposed an industrial application of a human–robot collaborative parallel two-sided destructive disassembly line balancing problem in multi-product and multi-line layouts [19]. Liang et al. developed a multi-product disassembly line balancing optimization method aiming at high disassembly profit and low energy consumption with noise pollution constraints [30].

These works confirm that the multi-product dimension is increasingly considered in disassembly line balancing. They also show that realistic industrial systems may require additional features such as destructive modes, environmental constraints, maintenance planning or parallel line structures. However, most of these contributions are not specifically focused on electric vehicle batteries or do not integrate stochastic human operation times. They are therefore useful for understanding the broader evolution of multi-product disassembly, but they do not fully cover the positioning of the present thesis.

2.8.4 Multi-Product HRC and Remaining Limitations

Although the multi-product dimension is important, it is still less developed than single-product disassembly planning in the battery literature. Many studies optimize the disassembly of one battery pack or one module, such as a Tesla, Audi or Volkswagen battery. These studies are useful because they provide detailed product-level models, but their results are strongly linked to the selected case study. When the battery type changes, the precedence graph, task durations and feasible execution modes may also change.

The work of Qin et al. represents an important step toward multi-product human-robot disassembly line balancing [39]. However, the combination of multi-product modeling, human-robot collaboration and uncertainty remains limited in the literature. In particular, many models still rely on deterministic task durations. This assumption simplifies the formulation, but it does not fully represent the variability of human disassembly operations in end-of-life battery treatment.

Another limitation concerns the objective structure. Existing studies often use several performance criteria such as number of stations, workload balance, cost, risk or energy. These criteria are relevant, but a multi-objective formulation may produce a set of compromise solutions that require an additional decision-making step. In contrast, the present work adopts a mono-objective economic formulation. This choice makes the decision criterion clearer by aggregating the main cost components into a single objective function. The deterministic version minimizes station opening and execution costs, while the stochastic version adds an expected recourse cost associated with station overloads under uncertain human operation times.

Therefore, multi-product disassembly lines represent a key research direction for realistic battery recovery systems. They make it possible to move from product-specific case studies toward a more general decision framework. The present research builds on this idea by considering several battery types in the same human-robot collaborative line. Each product keeps its own disassembly structure, but all products share the same potential stations and resources. This allows the model to represent a more flexible and industrially relevant disassembly system.

2.9 Synthesis of the Literature and Research Gap

The reviewed literature shows that disassembly planning for end-of-life electric vehicle batteries has become an active research topic. Existing studies address the problem from different angles, including disassembly sequence planning, disassembly line balancing, human-robot task allocation, robotic disassembly, stochastic modeling, risk-aware planning and energy-oriented optimization. These contributions provide a strong foundation for understanding how battery disassembly can be organized and optimized. However, they also show that several important dimensions are still rarely integrated within the same framework.

2.9.1 Main Trends Identified in the Literature

A first important trend is the use of structured product representations. Most studies rely on precedence graphs, hybrid disassembly graphs, priority matrices or relation matrices to represent the logical and physical constraints between components. Baazouzi et al. [3] used precedence relations to optimize disassembly strategy and disassembly depth. Ke et al. [27] used a hybrid graph and a relation matrix to represent battery component dependencies. Zhan et al. [53] relied on a hybrid disassembly graph for environment-oriented sequence planning. Yin et al. [51] introduced a knowledge graph and a graph neural network to represent richer relations between components, tools and operations. These works show that structured representations are essential for generating feasible disassembly plans.

A second trend concerns the increasing use of heuristic and metaheuristic methods. Battery disassembly planning problems are generally combinatorial and difficult to solve

exactly when the number of tasks increases. Wu et al. [48] used NSGA-II for a cell-level battery module disassembly line balancing problem. Chu and Chen [12] proposed an optimization approach for HRC battery disassembly planning. Guo et al. [21] used a multi-objective shuffled frog leaping algorithm for stochastic HRC disassembly line balancing. Zhu et al. [60] proposed a multi-objective adaptive large neighbourhood search algorithm for a stochastic sequence-dependent DLBP. Wu et al. [47] developed a hybrid evolutionary algorithm for a stochastic HRC disassembly line balancing problem. This observation is important for the present work, since the proposed mathematical model is expected to become difficult to solve exactly for larger multi-product instances. It therefore supports the development of a metaheuristic solution approach for the larger cases considered after the exact modelling phase.

A third trend is the growing interest in human-robot collaboration. Villagrossi and Dinon [44] showed that robotics can support battery pack disassembly but that full automation remains limited by variability. Kaarlela et al. [25] reached a similar conclusion in their review of robotised battery disassembly. Chu and Chen [12] explicitly modeled human-robot collaboration for end-of-life power batteries. Qin et al. [39] extended this direction to a multiple product hybrid disassembly line balancing problem with HRC. These works show that HRC is a realistic intermediate solution between fully manual and fully automated disassembly.

A fourth trend is the progressive consideration of uncertainty. Many early models assume deterministic operation times, costs and resource requirements. However, end-of-life battery disassembly is strongly affected by variability. Fu et al. [15] modeled stochastic processing times in an integrated remanufacturing scheduling problem. Yuan et al. [52] used fuzzy processing times when precise probability distributions were not available. Guo et al. [21] considered stochastic operation times in HRC disassembly line balancing. Zhu et al. [60] introduced stochastic and sequence-dependent effects in DLBP. Zhang et al. [57] studied stochastic task times and energy consumption in an energy-oriented DLBP. These works justify the transition from a deterministic formulation to a stochastic one when human operation times are uncertain.

A fifth trend is the expansion of performance criteria. Many articles consider time, cost, risk, energy, profit, environmental impact, remanufacturing demand or carbon emissions. This diversity shows that disassembly is a multi-dimensional problem. However, it also means that the decision-maker often needs to select one final solution among several compromises. The present thesis does not ignore this richness, but it chooses a mono-objective economic formulation in order to keep the model clear and operational.

2.9.2 Main Limitations

Despite these advances, several limitations remain. The first limitation is the predominance of single-product case studies. Most works focus on one specific battery pack or module, such as a Tesla, Audi or Volkswagen battery. This allows a detailed modeling of the selected product, but it does not fully represent industrial dismantling centers, where several battery types may be received and processed using the same resources. In practice, battery architectures vary significantly across manufacturers and vehicle models, which makes the multi-product dimension essential [26].

The second limitation is the frequent use of deterministic data. In many studies, operation times are assumed to be fixed and known in advance. This assumption is useful for building a first model, but it does not fully reflect the real variability of end-of-life batteries. Human disassembly times are especially uncertain because operators

must adapt to the actual condition of each battery. Therefore, a deterministic model can be considered as a necessary baseline, but it should be extended when the objective is to better represent real operating conditions.

The third limitation concerns the integration of human–robot collaboration with other realistic dimensions. Several works study HRC battery disassembly, but they often remain focused on one product type or deterministic operation times. On the other hand, some works introduce multi-product or stochastic aspects, but without fully integrating collaborative human–robot execution modes. As a result, the combination of multi-product disassembly, human–robot collaboration and stochastic human operation times remains insufficiently explored.

The fourth limitation is related to the objective structure. Many studies formulate the problem as a multi-objective optimization problem, considering criteria such as time, risk, energy, number of stations or cost. These criteria are relevant, but they may lead to a set of Pareto solutions that require an additional selection step. In contrast, an economic mono-objective formulation can be useful when the aim is to provide a clear operational decision. By aggregating station opening costs, task execution costs and, in the stochastic case, recourse costs, the decision-maker obtains a single criterion that remains interpretable from an industrial point of view.

2.9.3 Positioning of the Present Research

Based on these observations, the present research is positioned at the intersection of multi-product disassembly line planning, human–robot collaborative task allocation and stochastic modeling of human operation times. The first step of the work consists in developing a deterministic mono-objective optimization model. This model defines the basic structure of the problem by assigning tasks of several battery types to stations and execution modes, while respecting precedence constraints, resource compatibility and cycle time limitations.

The deterministic model is useful because it provides a clear and rigorous formulation of the system. It allows the main decisions to be represented in a structured way: opening stations, assigning human and robotic resources, selecting execution modes, and organizing tasks under precedence constraints. However, this deterministic version is only a first approximation of the real disassembly process. Since end-of-life battery disassembly is affected by uncertain human operation times, the model is then extended to a stochastic scenario-based formulation.

In the stochastic extension, the structural decisions remain common to all scenarios, while scheduling decisions and station overload recourse variables become scenario-dependent. This makes it possible to account for the variability of human disassembly times without completely changing the structure of the deterministic model. The stochastic formulation also introduces a bounded recourse mechanism, allowing limited station overloads under uncertainty while penalizing them economically.

Therefore, the contribution of this work is positioned around a progressive optimization framework for multi-product human–robot collaborative battery disassembly. The deterministic model provides the exact reference, the stochastic extension improves the representation of uncertain human operation times, and the combinatorial complexity of the problem motivates the use of a metaheuristic solution method for larger instances.

2.10 Conclusion

This chapter reviewed the main research works related to disassembly planning, electric vehicle battery disassembly, human–robot collaboration, uncertainty modelling, and multi-product disassembly lines. The literature shows that end-of-life battery disassembly is a complex decision-making problem due to the diversity of battery architectures, the presence of hazardous components, the need to respect precedence constraints, and the variability of operation times.

The reviewed studies also show that several approaches have been proposed to address this complexity. Disassembly sequence planning methods focus on determining feasible removal orders, while disassembly line balancing models extend this logic by assigning tasks to workstations under cycle-time and precedence constraints. More recent works introduce human–robot collaboration in order to combine human flexibility with robotic precision and safety. Other contributions consider uncertainty through stochastic, fuzzy, or scenario-based models, recognizing that task durations may vary significantly in real end-of-life disassembly operations.

However, the literature also indicates that several aspects remain difficult to integrate within a single decision framework. Many studies focus on one specific product type, whereas practical recovery systems may have to process several battery models. In addition, several models rely on deterministic operation times, although human execution times can vary depending on product condition, accessibility, operator adaptation, and unexpected disassembly difficulties. Finally, when the number of products, tasks, stations, resources, and scenarios increases, the resulting optimization problem becomes highly combinatorial.

These observations provide the background for the next chapter. Chapter 3 will build on the literature review by defining the considered problem, clarifying the research positioning, and introducing the methodological framework adopted in this work.

Proposed HRC-ALNS Methodology for the Multi-Type Human–Robot Collaborative Battery Disassembly Problem

3.1 Introduction

The previous chapter reviewed the main contributions related to disassembly planning, electric vehicle battery disassembly, human–robot collaboration, uncertainty modelling, and multi-product disassembly line balancing. It showed that end-of-life battery disassembly is a complex decision-making problem, especially when several battery types, heterogeneous task structures, human and robotic resources, and uncertain operation times are considered together.

In the first part of this work, the studied problem was formulated through exact mixed-integer linear programming models. These models are useful because they provide a rigorous mathematical representation of the disassembly line. They explicitly describe station opening decisions, task-to-station assignment, execution mode selection, resource allocation, precedence constraints, cycle-time limitations, and, in the stochastic case, recourse decisions under uncertain human operation times.

However, the computational difficulty of the problem increases rapidly with the number of battery types and disassembly tasks. When more products are considered, the model must handle a larger number of assignment variables, precedence relations, execution mode combinations, resource constraints, and scenario-dependent variables. As a result, exact optimization becomes increasingly expensive in terms of computation time. This limitation motivates the development of an approximate solution approach capable of producing good-quality solutions within reasonable computational time.

The purpose of this chapter is therefore to introduce a metaheuristic approach adapted to the multi-type human–robot collaborative battery disassembly problem. The selected framework is based on Adaptive Large Neighbourhood Search (ALNS). ALNS is a known metaheuristic in the literature; it is not introduced here as a new independent algorithm. In this work, the contribution lies in its adaptation to the considered problem.

The proposed adaptation concerns several specific elements of the problem: the internal encoding of a solution, the decoding procedure used to assign tasks to stations, the management of human, robotic and collaborative execution modes, the treatment of AND/OR precedence relations, and the design of destruction and repair operators adapted to both deterministic and stochastic cases.

For clarity, the proposed variant is referred to as HRC-ALNS, for Human–Robot Collaborative Adaptive Large Neighbourhood Search. Two versions are considered. The deterministic version is denoted D-HRC-ALNS, while the stochastic version is denoted

S-HRC-ALNS. The deterministic version uses nominal task durations and evaluates station loads under fixed operation times. The stochastic version incorporates scenario-dependent human operation times and evaluates the expected recourse generated by station overloads under uncertainty.

The chapter is organized as follows. Section 3.2 presents the research problem and the methodological positioning of the proposed approach. Section 3.3 introduces the general principle of ALNS and explains how it is adapted to the studied problem. Section 3.4 presents the internal encoding of a solution. Sections 3.5 and 3.6 describe the construction of the initial solution and the decoding procedure. Sections 3.7 and 3.8 present the neighbourhood generation mechanisms. Sections 3.9 and 3.10 explain the evaluation, acceptance criterion, and adaptive operator selection. Finally, Section 3.11 summarizes the general algorithmic framework.

3.2 Research Problem and Methodological Positioning

The problem addressed in this chapter concerns the resolution of a multi-type human–robot collaborative disassembly line balancing problem for end-of-life electric vehicle batteries. Several battery types are processed on the same line. Each battery type has its own set of tasks, precedence relations, processing times, feasible execution modes, and task characteristics. The line contains a maximum number of potential stations, and each task must be assigned to one station and executed by a compatible resource configuration.

The main decisions are interdependent. A task cannot be assigned independently of its precedence relations. A collaborative task cannot be scheduled unless both a human and a robot are available at the same station. A normal task may be executed either by a human or by a robot, which creates flexibility but also increases the number of possible combinations. In the stochastic case, the same structural assignment must remain valid under several scenarios of human operation times. These elements make the problem highly combinatorial.

The exact MILP formulations provide a strong reference framework. They define the constraints precisely and allow optimal solutions to be obtained for small instances. Nevertheless, when the number of products and tasks increases, the number of binary variables and constraints grows significantly. The stochastic formulation is even more demanding because scheduling variables, station loads and recourse variables are scenario-dependent. This explains why a complementary solution approach is needed for larger instances.

The research problem considered in this chapter can therefore be formulated as follows:

How can the deterministic and stochastic multi-type human–robot collaborative battery disassembly line problem be solved efficiently when exact optimization becomes computationally restrictive, while preserving the feasibility of task sequences, execution modes, AND/OR precedence relations, station assignments and recourse evaluation?

To answer this question, the present work adopts a problem-specific adaptation of ALNS. The objective is not to replace the mathematical model, but to use the structure of the model as a basis for designing a scalable search procedure. The exact formulation defines the logic of feasibility and evaluation, while the metaheuristic explores the search space more flexibly.

This methodological positioning is important. The proposed HRC-ALNS is not presented as a completely new metaheuristic. It remains based on the general ALNS principle: starting from a complete solution, removing part of it, repairing it in a different way, evaluating the new solution, and deciding whether to accept it. The added value of this work lies in the adaptation of these steps to the specific characteristics of the studied disassembly problem.

In particular, the proposed method must be able to handle:

- several battery types with different task sets and precedence relations;
- human, robotic and collaborative execution modes;
- AND and OR precedence relations;
- station assignment and resource allocation through a decoder;
- deterministic and stochastic evaluations;
- recourse costs in the stochastic case;
- neighbourhood operators that target bottleneck stations, long tasks, costly tasks, human variability and scenario-dependent overloads.

The following section introduces the transition from the general ALNS logic to the proposed HRC-ALNS variant.

3.3 From ALNS to the Proposed HRC-ALNS Variant

3.3.1 General Principle of Adaptive Large Neighbourhood Search

Adaptive Large Neighbourhood Search is a metaheuristic based on the iterative modification of complete solutions. At each iteration, a current solution is partially destroyed by removing a subset of elements. The incomplete solution is then repaired by reinserting the removed elements according to one or several repair rules. This destruction–repair mechanism allows the algorithm to explore large neighbourhoods of the current solution [?, ?].

The term adaptive refers to the fact that several destruction and repair operators can be used during the search. Their selection is not fixed once and for all. Instead, operator weights are updated according to their past performance. Operators that frequently lead to improved solutions receive higher selection probabilities, while less effective operators become less likely to be selected. This adaptive mechanism makes the search more flexible than a classical local search based on a single neighbourhood structure.

In many optimization problems, ALNS is combined with an acceptance criterion inspired by simulated annealing. A new solution is always accepted if it improves the current solution. If it is worse, it may still be accepted with a probability that decreases with the deterioration of the objective value and with the temperature parameter. This mechanism helps the algorithm avoid premature convergence to a local optimum.

For disassembly line balancing problems, the ALNS principle is particularly relevant because the solution is composed of several interdependent decisions. Modifying a small part of the solution can affect station loads, precedence feasibility, resource availability and execution cost. Recent works have also used ALNS-based approaches for stochastic and sequence-dependent disassembly line balancing, which confirms the interest of this search logic for complex end-of-life product disassembly problems [60].

3.3.2 Motivation for Adapting ALNS to the Studied Problem

The studied problem has a structure that is well suited to an ALNS-based approach. First, a complete solution can be represented by a sequence of operations and a set of execution mode decisions. Second, part of this solution can be removed without discarding the entire solution. Third, the removed tasks can be reinserted in different feasible positions, allowing the search to explore new task orders, mode combinations and station assignments.

However, a direct application of a standard ALNS is not sufficient. The studied problem includes several characteristics that require a specific adaptation. The first one is the presence of several battery types. A solution must include all tasks of all products, while preserving the internal precedence structure of each battery type. The second one is the presence of human, robotic and collaborative execution modes. A task may have only one feasible mode or may allow several alternatives. The third one is the presence of AND/OR precedence relations, which means that some predecessors are mandatory while others are selected among alternatives.

Another important point is the station assignment. In the proposed method, stations are not directly encoded in the main solution representation. Instead, they are computed by a decoder after each modification of the sequence and execution modes. This choice avoids generating too many infeasible solutions during the search. It also separates two roles: the encoding is used to explore the search space, while the decoder translates the encoded solution into stations, resource allocations, loads, recourse values and final cost.

The stochastic version requires an additional adaptation. In this case, human operation times vary across scenarios. Therefore, the evaluation of a solution cannot rely only on nominal station loads. The decoder must calculate scenario-dependent station loads and the possible recourse associated with overloads. Destruction and repair operators must also exploit stochastic information, such as stations with high recourse, tasks with high human-time variability, and tasks contributing to the worst scenario.

3.3.3 Definition of the Proposed HRC-ALNS Variants

Based on these adaptations, the proposed approach is denoted HRC-ALNS, for Human-Robot Collaborative Adaptive Large Neighbourhood Search. This name reflects the fact that the algorithm is not a generic ALNS applied without modification, but a variant designed for a human-robot collaborative disassembly line problem.

Two versions are distinguished:

- **D-HRC-ALNS:** the deterministic version, where all operation times are fixed and the evaluation is based on nominal station loads, station opening cost, execution cost and feasibility penalties;
- **S-HRC-ALNS:** the stochastic version, where human operation times depend on scenarios and the evaluation includes expected recourse costs associated with station overloads.

Both versions share the same general structure. A solution is encoded by a sequence of operations, a set of execution modes, and the active OR choices. The solution is then decoded into stations and resources. At each iteration, a mixed destruction mechanism removes a subset of tasks, and a hybrid repair mechanism reinserts them in feasible positions. The resulting solution is decoded, evaluated and either accepted or rejected according to the acceptance rule.

The main difference between the two versions lies in the information used by the operators and by the evaluation function. In D-HRC-ALNS, destruction and repair are guided by nominal loads, long tasks, costly tasks and bottleneck stations. In S-HRC-ALNS, they also consider scenario-dependent recourse, variability of human task durations and worst-scenario station loads. This distinction allows the stochastic version to search not only for low-cost solutions, but also for solutions that are less sensitive to human-time variability.

Table 3.1 summarizes the main components of the proposed HRC-ALNS methodology.

Table 3.1: Main components of the proposed HRC-ALNS methodology.

Component	Role in the proposed methodology
Solution encoding	Represents a candidate solution through the operation sequence, execution modes and OR choices.
Decoder	Converts the encoded solution into station assignments, resource allocations, station loads, recourse values and total cost.
Mixed destruction	Removes a subset of tasks according to several criteria in order to diversify the search.
Hybrid repair	Reinserts removed tasks using rules adapted to the reason for their removal and to the deterministic or stochastic case.
Solution evaluation	Computes deterministic or stochastic objective values, including penalties when required.
Acceptance rule	Uses simulated annealing to accept improving solutions and occasionally accept non-improving ones.
Adaptive selection	Updates operator weights according to their contribution to the search.

3.4 Solution Representation

The efficiency of a metaheuristic strongly depends on the way candidate solutions are represented. In the studied problem, a solution must contain enough information to describe the order of operations, the execution mode of each task, and the selected alternatives in the case of OR precedence relations. At the same time, the representation must remain simple enough to be modified by destruction and repair operators during the search.

For this reason, the proposed HRC-ALNS does not encode station assignments directly. Instead, a solution is represented internally by a sequence of operations, a set of execution mode decisions, and a set of active OR choices. The assignment to stations is obtained later through a decoding procedure. This separation between encoding and decoding makes the search more flexible and reduces the risk of generating infeasible solutions after each neighbourhood move.

3.4.1 Internal Encoding of a Solution

Let an operation be denoted by (p, i) , where $p \in P$ represents the battery type and $i \in I_p$ represents a disassembly task of product p . In the proposed HRC-ALNS, a solution is encoded as follows:

$$S = (\pi, \mu, z), \quad (3.1)$$

where:

- π is the global sequence of operations. It contains all operations (p, i) belonging to all battery types considered in the instance;
- μ is the execution mode vector. It specifies the selected execution mode for each operation;
- z represents the active OR choices. For tasks with alternative predecessors, it indicates which predecessor is selected.

The sequence π defines the order in which operations are considered by the decoder. It is not a direct schedule, because start times and station assignments are not stored in the sequence. Instead, π provides a priority order used to build a feasible line configuration. Each operation must appear exactly once in π , which ensures that no disassembly task is omitted or duplicated.

The execution mode vector μ associates each operation (p, i) with one feasible mode:

$$\mu_{pi} \in \{H, R, C\}, \quad (3.2)$$

where H denotes human execution, R denotes robotic execution, and C denotes collaborative human–robot execution. However, the feasible set of modes depends on the nature of the task. Human-only tasks can only be assigned to H , robot-only tasks can only be assigned to R , and collaborative tasks can only be assigned to C . Normal tasks can be assigned either to H or to R .

The vector z is used only for tasks that have OR precedence relations. In such cases, a task may become feasible after one predecessor among several alternatives has been completed. Therefore, z stores the selected active predecessor for each task concerned by an OR relation. This choice is important because it determines the precedence constraints that must be respected during repair and decoding.

3.4.2 Encoding of Execution Modes

Execution modes play a central role in the proposed method because they connect the disassembly task to the required resources. A task executed in human mode requires a human operator at the assigned station. A task executed in robot mode requires a robotic resource. A collaborative task requires both resources at the same station and during the same operation.

The mode decision therefore affects several elements of the solution evaluation. It influences the processing time, the execution cost, the type of station required, the station load, and the feasibility of resource allocation. For this reason, mode changes are important neighbourhood moves in the HRC-ALNS. In particular, normal tasks provide flexibility because their execution mode can be modified between human and robot execution during the search.

In the deterministic version, the selected mode determines the fixed processing time used to evaluate the station load. In the stochastic version, the selected mode also determines whether the task is affected by uncertainty. Human operation times vary across scenarios, while robot and collaborative operation times remain deterministic in the adopted formulation.

3.4.3 Encoding of OR Precedence Choices

The studied disassembly problem includes both AND and OR precedence relations. AND precedence relations are mandatory. If task i is an AND-predecessor of task j , then task i must be completed before task j according to the technological logic of the disassembly process.

OR precedence relations are different. They represent alternative access conditions. If a task has several possible OR-predecessors, the completion of one selected predecessor is sufficient to activate the task. In the proposed encoding, this selection is stored in the vector z .

For a task (p, j) with a set of alternative predecessors $Pred_{pj}^{OR}$, the selected OR predecessor is represented as:

$$z_{pj} \in Pred_{pj}^{OR}. \quad (3.3)$$

During repair, the selected OR choices are used to determine feasible insertion positions in the sequence. A task cannot be inserted before its active predecessors, and it cannot be placed in a position that violates the order of its active successors. During decoding, the same active precedence logic is used to determine the earliest station where the task can be assigned.

This treatment allows the algorithm to preserve the logic of the disassembly process while keeping the solution representation compact. It also allows OR choices to be modified during the search when such a modification can improve feasibility, cost, workload balance, or recourse performance.

3.4.4 Why Stations Are Not Directly Encoded

A possible representation would consist of encoding, for each task, its station assignment together with its execution mode. However, this direct encoding would make the search more difficult. A simple modification of the sequence or a mode could easily create infeasible station assignments, for example by violating precedence relations, exceeding the cycle time, opening unnecessary stations, or assigning a collaborative task to a station without both resources.

For this reason, the proposed HRC-ALNS separates the internal representation from the final line configuration. The solution manipulated by the metaheuristic contains only the sequence, execution modes, and OR choices. The station assignment is reconstructed by a decoder after each modification.

This design has two advantages. First, the destruction and repair operators can modify the solution without directly managing all station feasibility constraints. Second, the decoder can evaluate each candidate solution consistently by applying the same assignment logic. It computes the opened stations, the resource configuration of each station, the station loads, the recourse values in the stochastic case, and the final objective value.

In other words, the encoding is used to explore the search space, while the decoder translates the encoded solution into an operational disassembly line configuration. Figure 3.1 summarizes this principle.

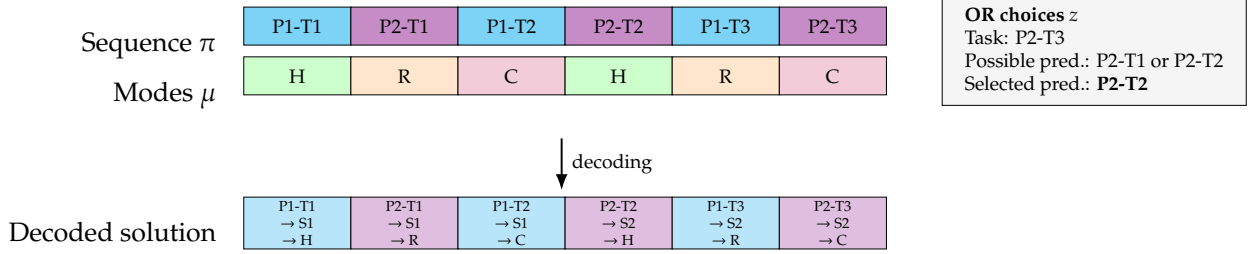


Figure 3.1: Internal encoding of a solution and decoding toward stations.

3.5 Initial Solution Construction

The HRC-ALNS starts from a complete initial solution. This solution does not need to be optimal, but it must provide a feasible or nearly feasible starting point for the search. The initial solution is built in three main steps: selection of OR predecessors, construction of a precedence-feasible operation sequence, and assignment of execution modes.

First, for each task having OR precedence relations, one active predecessor is selected. When no additional information is available, this choice can be made according to a simple rule, such as selecting the predecessor with the shortest processing time, the lowest execution cost, or a random predecessor among feasible alternatives. The selected OR predecessors define the active precedence structure used by the sequence construction procedure.

Second, the algorithm constructs an initial sequence π containing all operations of all battery types. The construction follows a precedence-feasible logic. At each step, the set of available operations is determined. An operation is available if all its active predecessors have already been inserted into the sequence. One operation is then selected from the available set and appended to π . This process continues until all operations have been inserted exactly once.

Third, an initial execution mode is assigned to each operation. For tasks with a single feasible mode, the assignment is direct. Human-only tasks are assigned to H , robot-only tasks to R , and collaborative tasks to C . For normal tasks, which can be executed either by a human or by a robot, the initial mode can be selected using a simple constructive rule. For example, the algorithm may choose the mode with the shortest processing time, the lowest execution cost, or the mode that improves the expected load balance.

After these three steps, the encoded solution $S = (\pi, \mu, z)$ is passed to the decoder. The decoder assigns operations to stations, determines the required resources, computes station loads and evaluates the objective function. If the decoded solution violates some constraints, a penalty is added to its objective value. The search can then start from this initial solution and progressively improve it through destruction and repair moves.

3.6 Decoding Procedure

The decoder is one of the most important components of the proposed methodology. It transforms the encoded solution $S = (\pi, \mu, z)$ into an operational line configuration. The

output of the decoder includes station assignments, station openings, human and robotic resource allocations, station loads, recourse values in the stochastic case, and the final objective value.

3.6.1 Precedence-Based Feasibility

The decoder reads the operations in the order defined by π . For each operation (p, i) , it identifies the active predecessor set. This set contains all mandatory AND predecessors and the selected OR predecessors stored in z . It is denoted by:

$$Pred_{pi}^+ = Pred_{pi}^{AND} \cup Pred_{pi}^{OR,z}. \quad (3.4)$$

If a predecessor of (p, i) has already been assigned to a station, the current operation cannot be placed in a station located before that predecessor. Therefore, the earliest admissible station is computed as:

$$k_{\min}(p, i) = \max\{k(p, h) : h \in Pred_{pi}^+\}, \quad (3.5)$$

where $k(p, h)$ is the station assigned to predecessor h of product p . If the operation has no active predecessor, then $k_{\min}(p, i) = 1$.

This rule ensures that the physical order of stations does not violate the active precedence structure. If two tasks are assigned to the same station, the decoder also preserves the order induced by the sequence and by the internal scheduling procedure.

3.6.2 Station Assignment and Resource Allocation

After computing the earliest admissible station, the decoder tests candidate stations $k \geq k_{\min}(p, i)$. A station is considered compatible with an operation if it can provide the resources required by the selected execution mode. If a task is executed in mode H , the station must contain a human operator. If it is executed in mode R , the station must contain a robot. If it is executed in mode C , both resources must be available.

If a compatible opened station exists and can receive the operation without excessive overload, the task is assigned to that station. Otherwise, the decoder may open a new station, provided that the maximum number of potential stations is not exceeded. The opening of a station generates a station cost and determines whether the station is human-only, robot-only, or collaborative.

The decoder therefore plays a constructive role. It does not simply read the encoded solution; it converts it into a feasible or penalized line configuration by applying the same logic as the mathematical model: precedence feasibility, resource compatibility, station opening and workload evaluation.

3.6.3 Deterministic Decoding

In the deterministic version, operation times are fixed. The decoder uses the same assignment indicators as the mathematical model. For each operation (p, i) and station k , x_{pik}^H , x_{pik}^R and x_{pik}^C indicate whether the task is assigned to station k and executed respectively in human, robot or collaborative mode. The nominal load of station k is therefore written as:

$$L_k = \sum_{p \in P} \sum_{i \in I_p} (t_{pi}^H x_{pik}^H + t_{pi}^R x_{pik}^R + t_{pi}^C x_{pik}^C). \quad (3.6)$$

The main capacity condition is:

$$L_k \leq CT. \quad (3.7)$$

If the decoded assignment satisfies this condition for all opened stations, no overload penalty is required. If a station exceeds the cycle time, the solution is penalized. This penalty does not mean that the solution is accepted as feasible in the final decision. It is used during the search to guide the algorithm away from infeasible regions.

3.6.4 Stochastic Decoding and Recourse Evaluation

In the stochastic version, the same station assignment and execution mode decisions are evaluated over all scenarios. Human operation times vary according to the scenario, while robotic and collaborative times remain deterministic. For each station k and scenario s , the decoder computes a scenario-dependent load:

$$L_k^s = \sum_{p \in P} \sum_{i \in I_p} (t_{pi}^{H,s} x_{pik}^H + t_{pi}^R x_{pik}^R + t_{pi}^C x_{pik}^C). \quad (3.8)$$

The overload or recourse of station k in scenario s is defined as:

$$r_k^s = \max(0, L_k^s - CT). \quad (3.9)$$

The recourse must remain bounded according to the admissible tolerance:

$$r_k^s \leq \eta CT u_k, \quad \eta = 0.15, \quad (3.10)$$

where u_k indicates whether station k is opened and η is the maximum allowed overload ratio. The same value adopted in the stochastic MILP is used, namely $\eta = 0.15$.

The stochastic decoder is therefore more demanding than the deterministic one. It evaluates each candidate assignment not only under a nominal duration, but under all generated scenarios. This makes it possible to detect solutions that appear balanced in nominal conditions but become weak under human-time variability.

3.7 Neighbourhood Generation by Mixed Destruction

The destruction phase is responsible for creating a partial solution by removing a subset of tasks from the current solution. This step is essential because it allows the algorithm to escape the current local structure and explore alternative configurations. In the proposed HRC-ALNS, the destruction mechanism is mixed: the removed tasks are selected according to several criteria rather than by a single rule applied to the whole removed set.

3.7.1 Destruction Size

At each iteration, the number of removed tasks is not fixed. It depends on the total number of operations in the current instance. Let n be the number of operations included in the solution. The destruction size is defined as:

$$q = \max(1, \text{round}(U(0.10, 0.35) \times n)), \quad (3.11)$$

where $U(0.10, 0.35)$ is a uniform random draw between 10% and 35%.

This variable destruction size allows the algorithm to alternate between intensification and diversification. Small destructions modify only a limited part of the solution and are useful for local improvement. Larger destructions modify a more significant part of the solution and help the search move away from poor regions.

3.7.2 Deterministic Destruction Operators

In the deterministic version, the destruction phase uses nominal durations, nominal station loads and execution costs. Table 3.2 summarizes the deterministic destruction operators.

Table 3.2: Deterministic destruction operators used in D-HRC-ALNS.

Operator	Role
Random removal	Removes tasks randomly in order to maintain diversification during the search.
Bottleneck station removal	Removes tasks assigned to the station with the highest nominal load.
Long task removal	Removes tasks with large processing times, since they strongly influence cycle-time feasibility.
Costly task removal	Removes tasks with high execution cost in order to allow a different mode or station assignment.
Normal task removal	Removes normal tasks that can be executed either by a human or by a robot, making mode changes possible.

These operators target different weaknesses of the current solution. Bottleneck removal acts on workload imbalance. Long task removal focuses on tasks that constrain the cycle time. Costly task removal seeks to reduce the objective value. Normal task removal exploits mode flexibility between human and robot execution.

3.7.3 Stochastic Destruction Operators

In the stochastic version, the destruction phase uses additional information obtained from the scenario evaluation. The objective is not only to reduce nominal cost, but also to improve the behaviour of the solution under uncertainty. Table 3.3 presents the stochastic destruction operators.

Table 3.3: Stochastic destruction operators used in S-HRC-ALNS.

Operator	Role
Random removal	Maintains diversification even in the stochastic search.
Recourse station removal	Removes tasks from stations generating high recourse over the scenarios.
High variability human removal	Removes human tasks whose processing times vary strongly between scenarios.
Worst-scenario task removal	Removes tasks contributing to the highest station load in the most unfavorable scenario.
Recourse-risk removal	Combines recourse information and human-time variability to target tasks that create overload risk.

The expression risk in this context refers to the risk of exceeding the cycle time under uncertain human operation times. It does not necessarily indicate a physical safety risk. A task is considered risky when its duration variability can contribute to station overload in one or several scenarios.

3.7.4 Mixed Destruction Strategy

In a classical ALNS implementation, one destruction operator is often selected and applied to the solution. In the proposed HRC-ALNS, the destruction is mixed. Once the total destruction size q is selected, this quantity is distributed across several criteria. For example, in the deterministic case, part of the removed tasks may come from the bottleneck station, another part from the longest tasks, and another part from normal tasks. In the stochastic case, the removed set may combine tasks from high-recourse stations, highly variable human tasks and tasks contributing to the worst scenario.

This mixed strategy is more adapted to the studied problem because a poor solution may have several causes at the same time. A station may be overloaded, some tasks may be expensive, and some flexible tasks may have inappropriate execution modes. In the stochastic case, the solution may also be weak because of human-time variability. Removing tasks according to several criteria gives the repair mechanism more opportunities to reconstruct a better configuration.

Figure 3.2 illustrates the principle of mixed destruction.

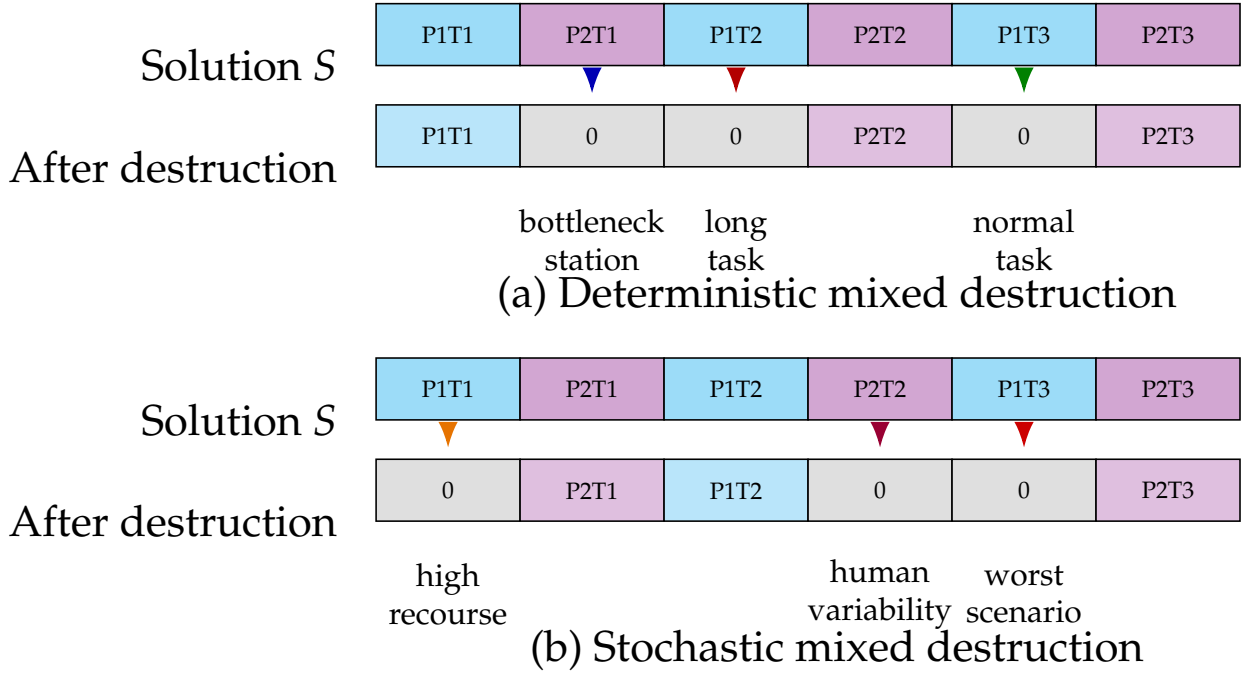


Figure 3.2: Example of mixed destruction: several criteria contribute to the removed subset.

3.8 Hybrid Repair Mechanism

After destruction, the solution becomes partial. Some tasks are temporarily absent from the sequence and must be reinserted. The repair mechanism reconstructs a complete solution by inserting each removed task into a feasible position. In the proposed method, the repair is hybrid because the rule used for reinsertion depends on both the type of case and the reason why the task was removed.

3.8.1 Feasible Insertion Positions

Before reinserting a removed operation (p, i) , the algorithm identifies the positions in the partial sequence where the insertion is feasible. A position is feasible if all active predecessors of (p, i) are located before the position and all active successors are located after it. The active predecessors include the AND predecessors and the OR predecessor selected in z .

This step is necessary because inserting a task at a random position can easily violate the disassembly logic. By computing feasible positions first, the repair mechanism preserves the technological consistency of the sequence before the decoder assigns tasks to stations.

It is important to distinguish two levels of feasibility. The repair mechanism verifies precedence feasibility inside the sequence. The decoder then verifies station assignment, resource compatibility, cycle-time limits and recourse constraints. This separation keeps the repair process manageable while allowing the decoder to evaluate the final operational feasibility of the solution.

3.8.2 Deterministic Repair Rules

In D-HRC-ALNS, repair rules are based on nominal durations, nominal loads and deterministic execution costs. Table 3.4 summarizes the main deterministic repair rules.

Table 3.4: Deterministic repair rules used in D-HRC-ALNS.

Repair rule	Role
Random repair	Reinserts a removed task into a feasible position selected randomly.
Sequential repair	Reinserts tasks according to a topological or natural order respecting precedence.
Least-cost repair	Selects the insertion and mode that reduce execution cost.
Fastest repair	Favors insertions and modes associated with short processing times.
Load-balanced repair	Selects the insertion that improves the balance of nominal station loads.
Minimum station opening repair	Favors insertions that avoid opening an additional station.

These rules are not used in isolation from the destruction phase. If a task was removed from a bottleneck station, load-balanced repair becomes particularly relevant. If a task was removed because it is long, fastest repair may be preferred. If a task was removed because it is costly, least-cost repair becomes more appropriate.

3.8.3 Stochastic Repair Rules

In S-HRC-ALNS, the repair mechanism must consider the behaviour of the candidate insertion over all scenarios. A position that seems good in nominal conditions may increase recourse in a scenario where human tasks become longer. Therefore, stochastic repair rules use scenario-dependent station loads and recourse information. Table 3.5 summarizes these rules.

Table 3.5: Stochastic repair rules used in S-HRC-ALNS.

Repair rule	Role
Random repair	Maintains a minimum level of diversification.
Recourse-aware repair	Tests feasible positions and selects the one that reduces expected recourse.
Robust repair	Favors insertions that reduce station load in the worst scenario.
Variability-aware repair	Avoids grouping several highly variable human tasks in the same station.
Scenario load-balanced repair	Balances station loads over all scenarios, not only under nominal conditions.

The stochastic repair rules make the S-HRC-ALNS different from a deterministic search followed by a stochastic evaluation. In the proposed approach, stochastic information is used during the construction of neighbouring solutions. This allows the search to avoid configurations that are likely to generate recourse, instead of discovering this weakness only after full evaluation.

3.8.4 Coupling Between Destruction and Repair

The repair phase is coupled with the destruction phase. This means that the cause of removal is stored and used to guide reinsertion. Table 3.6 gives examples of this coupling.

Table 3.6: Examples of coupling between destruction causes and repair rules.

Case	Removed tasks	Preferred repair logic
Deterministic	Tasks from bottleneck station	Load-balanced repair
Deterministic	Long tasks	Fastest repair
Deterministic	Costly tasks	Least-cost repair
Deterministic	Normal H/R tasks	Least-cost or mode-change repair
Stochastic	Tasks from high-recourse station	Recourse-aware repair
Stochastic	Highly variable human tasks	Variability-aware repair
Stochastic	Worst-scenario tasks	Robust repair
Stochastic	Tasks contributing to scenario imbalance	Scenario load-balanced repair

Figure 3.3 illustrates the principle of hybrid repair after destruction.

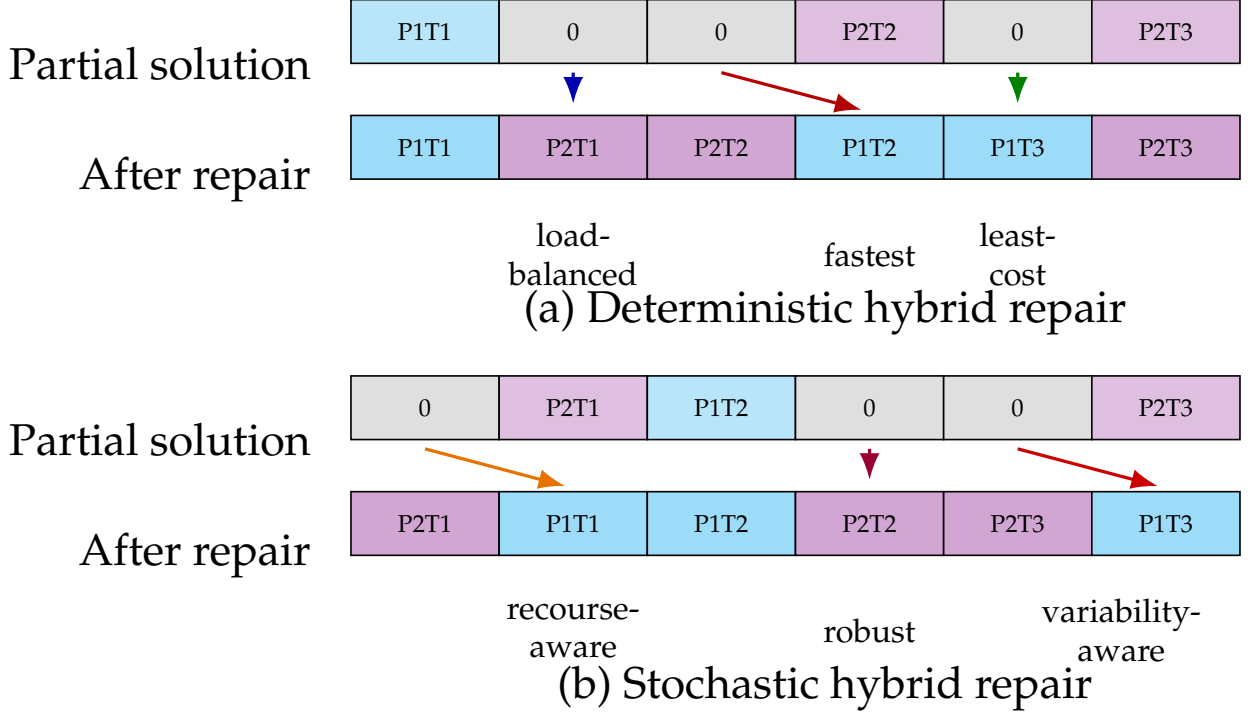


Figure 3.3: Example of hybrid repair after mixed destruction.

3.9 Evaluation of Candidate Solutions

After repair, the candidate solution is complete again. It is decoded and evaluated. The evaluation must be consistent with the optimization logic of the original mathematical model. For this reason, the objective function includes station opening costs, execution costs and, in the stochastic case, expected recourse costs. Penalty terms are added when the decoded solution violates important feasibility conditions.

3.9.1 Deterministic Evaluation

In D-HRC-ALNS, the candidate solution is evaluated using the same cost components as the deterministic mathematical formulation. The evaluation function is written as:

$$F_{det} = C_{stations} + C_{execution} + Penalty. \quad (3.12)$$

The station opening cost is computed as:

$$C_{stations} = \sum_{k \in K} f_k u_k, \quad (3.13)$$

where f_k is the fixed cost of opening station k and u_k indicates whether this station is used. The execution cost is computed from the selected execution modes:

$$C_{execution} = \sum_{p \in P} \lambda_p \sum_{i \in I_p} \sum_{k \in K} (c_{pi}^H x_{pik}^H + c_{pi}^R x_{pik}^R + c_{pi}^C x_{pik}^C). \quad (3.14)$$

The penalty term is used to discourage incomplete or infeasible solutions during the search. Typical sources of penalty include missing tasks, duplicated tasks, precedence

violations, cycle-time violations and incompatible resource assignments. The final reported solution must satisfy the feasibility conditions; penalties are used only as a search guidance mechanism.

3.9.2 Stochastic Evaluation

In S-HRC-ALNS, the same station and execution cost components are kept, but the evaluation also includes the weighted recourse cost over the generated scenarios. The stochastic evaluation function is written as:

$$F_{sto} = C_{stations} + C_{execution} + C_{recours} + Penalty. \quad (3.15)$$

The recourse cost is computed as:

$$C_{recours} = \sum_{s \in S} \omega_s \sum_{k \in K} \rho_k r_k^s, \quad (3.16)$$

with:

$$r_k^s = \max(0, L_k^s - CT). \quad (3.17)$$

Here, ω_s is the probability of scenario s , ρ_k is the unit recourse cost associated with station k , and r_k^s is the overload of station k in scenario s . When the same recourse cost is used for all stations, ρ_k can be replaced by a common coefficient c_R .

This evaluation gives the stochastic algorithm a different search direction from the deterministic version. A solution with a low deterministic cost may not be good under uncertainty if it produces frequent or high station overloads. Conversely, a solution that slightly increases station cost or execution cost may be more attractive if it significantly reduces weighted recourse.

3.9.3 Penalty Management

Penalty management is required because the destruction and repair operators may sometimes generate solutions that are incomplete or difficult to decode feasibly. Instead of immediately discarding all such solutions, the algorithm assigns a penalty to guide the search. The penalty increases the objective value and makes infeasible solutions less likely to survive.

The penalty mechanism is particularly useful during early iterations, when the algorithm explores a wide search space. However, the final reported solution must satisfy the problem constraints. Therefore, penalties are used as a search guidance mechanism, not as a substitute for feasibility.

3.10 Acceptance Criterion and Adaptive Operator Selection

3.10.1 Simulated Annealing Acceptance Rule

After a candidate solution S' is decoded and evaluated, it is compared with the current solution $S_{current}$. If the new solution is better, it is accepted directly. If it is worse, it may still be accepted with a probability inspired by simulated annealing:

$$P_{accept} = \exp\left(-\frac{F(S') - F(S_{current})}{T}\right), \quad (3.18)$$

where F denotes the deterministic or stochastic evaluation function and T is the current temperature.

The temperature decreases progressively during the search according to:

$$T \leftarrow \alpha T, \quad (3.19)$$

where α is the cooling rate. At the beginning of the search, the temperature is higher, and worse solutions can be accepted more easily. As the search progresses, the temperature decreases, and the algorithm becomes more selective. This mechanism improves exploration at the beginning and intensification near the end.

3.10.2 Operator Scoring and Weight Updating

The adaptive component of ALNS is based on the update of operator weights. Each destruction and repair operator has an associated weight. At the beginning, all operators may receive the same weight. During the search, operators that contribute to good solutions receive rewards, which increases their probability of being selected later.

Let w_o be the weight of operator o . After a given number of iterations, the weight can be updated as follows:

$$w_o \leftarrow (1 - \lambda)w_o + \lambda \frac{score_o}{use_o}, \quad (3.20)$$

where λ is the reaction factor, $score_o$ is the accumulated score obtained by the operator, and use_o is the number of times the operator was used during the update period.

The score assigned to an operator depends on the quality of the solution it helped generate. A high reward is given when the new solution improves the best-known solution. A smaller reward may be given when the new solution improves the current solution or is accepted by the simulated annealing criterion. Operators that do not contribute to accepted or improved solutions receive little or no reward.

This mechanism does not eliminate less effective operators completely. They may still be selected, which preserves diversification. However, the search progressively favors the operators that perform better for the instance being solved.

3.11 General Algorithmic Framework

Figure 3.4 summarizes the complete workflow of the proposed HRC-ALNS. The process starts with the construction and decoding of an initial solution. Then, at each iteration, a neighbouring solution is generated through mixed destruction and hybrid repair, decoded, evaluated, accepted or rejected, and used to update the best solution and the adaptive operator weights.

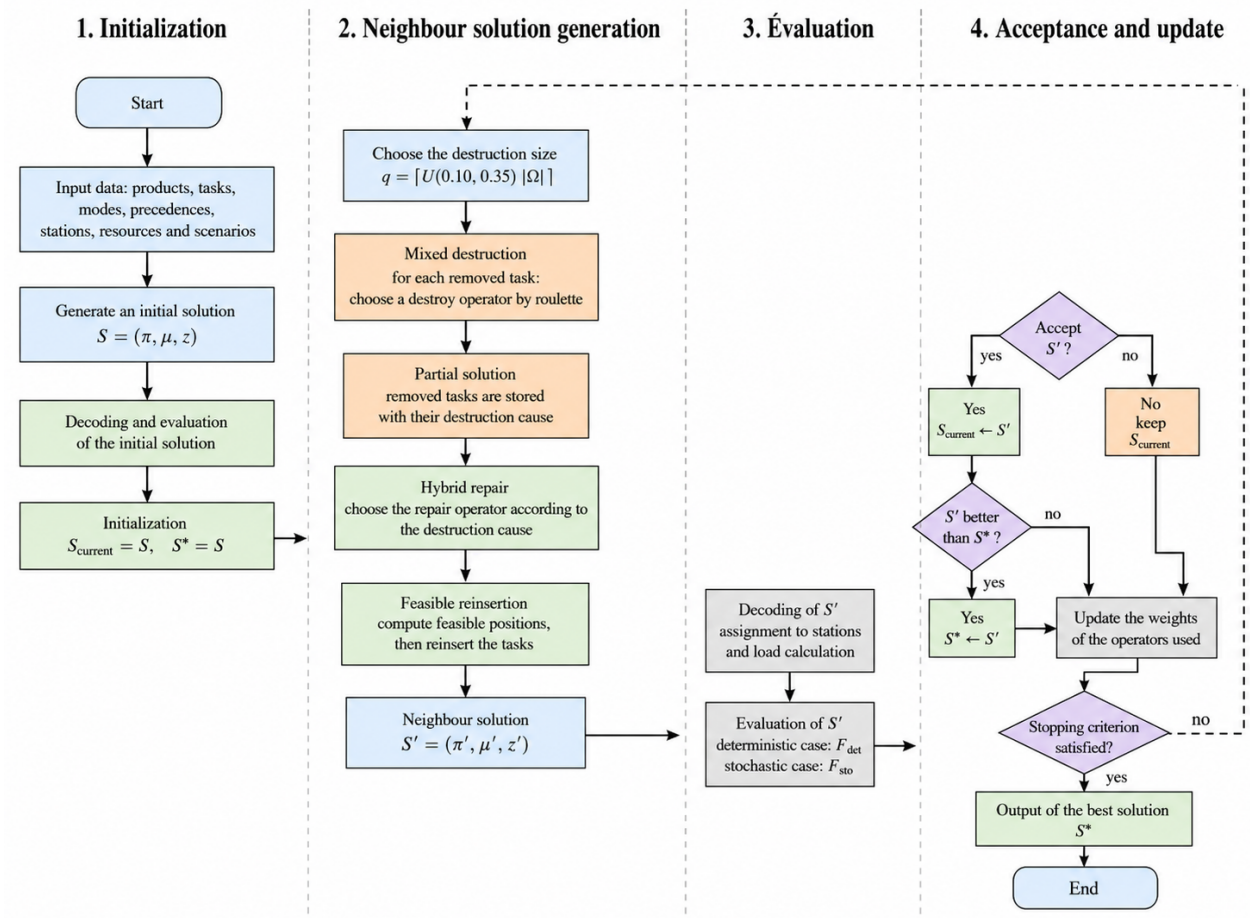


Figure 3.4: General workflow of the proposed HRC-ALNS methodology.

The detailed pseudo-code of the proposed method is given in Algorithm 1. It follows the same logic as the workflow shown in Figure 3.4, but it describes the operational steps in a more explicit way. Let $\Omega = \{(p, i) : p \in P, i \in I_p\}$ denote the set of all operations to be assigned. The detailed procedure of the proposed HRC-ALNS is presented in Algorithm 1. The algorithm follows the general logic of Adaptive Large Neighbourhood Search, where a complete solution is iteratively modified through destruction and repair mechanisms. In this work, this general structure is adapted to the considered multi-type human-robot collaborative disassembly problem. The procedure starts by reading the problem data, initializing the operators, constructing an initial solution, and decoding it into a complete line configuration.

Algorithm 1: General initialization of the HRC-ALNS procedure

Input:

- product, task, station and resource data;
- AND/OR precedence relations;
- feasible execution modes H, R and C ;
- cycle time CT ;
- scenarios and probabilities ω_s in the stochastic case;

- recourse parameters ρ_k and η ;
- destruction limits $\alpha_{\min} = 0.10$ and $\alpha_{\max} = 0.35$;
- initial temperature T_0 ;
- cooling rate β ;
- stopping threshold N_{stop} .

Output: best solution found S^* .

Begin

1. Read the problem data. If the case is stochastic, read the scenario data from the Excel input file.
2. Initialize the operator weights:
 $w_D = 1$ for each destruction operator D , $w_R = 1$ for each repair operator R .
3. Generate an initial solution $S = (\pi, \mu, z)$:
 - select the active OR predecessors z ;
 - generate a sequence π respecting the active precedence relations;
 - choose one admissible execution mode μ for each operation.
4. Decode S :
 - assign operations to stations;
 - respect the physical order of stations;
 - determine the type of each opened station;
 - compute station loads.
5. Evaluate S and set:

$$S_{current} = S, \quad S^* = S, \quad F_{best} = F(S),$$

$$iter = 0, \quad last_improvement_iter = 0, \quad T = T_0.$$

After initialization, the algorithm enters its iterative search process. At each iteration, the current solution is partially modified. The first phase is the mixed destruction phase. Its objective is to remove a limited number of operations from the current sequence in order to create room for improvement. The number of removed operations is randomly selected between the minimum and maximum destruction ratios. Several destruction operators may be used, and their selection depends on their adaptive weights.

Algorithm 1 continued: Phase 1 – Mixed destruction

6. While $iter - last_improvement_iter < N_{stop}$, repeat the following steps:
 - (a) Set $iter \leftarrow iter + 1$.

(b) Choose the number of operations to remove:

$$q = \lceil U(\alpha_{\min}, \alpha_{\max}) |\Omega| \rceil.$$

(c) Initialize:

$$\pi_{\text{partial}} = \pi_{\text{current}}, \quad \text{Removed} = \emptyset,$$

and initialize the destruction-use counter *DestroyUsage*.

(d) For $r = 1, \dots, q$:

- i. select a destruction operator D using roulette-wheel selection based on the current weights;
- ii. apply D to select one operation o to remove;
- iii. if D does not find any candidate operation, use random removal as a fallback rule;
- iv. remove o from π_{partial} ;
- v. store the removed operation and its cause:

$$\text{Removed} \leftarrow \text{Removed} \cup \{(o, D_{\text{used}})\};$$

vi. update *DestroyUsage*.

The second phase is the hybrid repair phase. Once a subset of operations has been removed, the algorithm reinserts them into the partial sequence. The repair process is not purely random. It uses the cause of destruction to select a suitable repair strategy. For example, operations removed from a bottleneck station are preferably reinserted using a load-balancing rule, while costly operations are reinserted using a cost-oriented rule. In the stochastic case, the repair phase also considers recourse and variability information.

Algorithm 1 continued: Phase 2 – Hybrid repair

(e) Sort the removed operations according to a feasible topological order.

(f) Initialize:

$$\pi_{\text{repaired}} = \pi_{\text{partial}},$$

and initialize the repair-use counter *RepairUsage*.

(g) For each removed operation o :

- i. identify the destruction cause of o ;
- ii. determine the repair operators compatible with this cause:
 - bottleneck station $\rightarrow$ load-balanced repair;
 - long task $\rightarrow$ fastest repair;
 - costly task $\rightarrow$ least-cost repair;
 - normal H/R task $\rightarrow$ repair with possible mode change;
 - station with recourse $\rightarrow$ recourse-aware repair;
 - variable human task $\rightarrow$ robust or variability-aware repair;
 - worst-scenario task $\rightarrow$ robust repair.
- iii. select a repair operator R by roulette-wheel selection among the compatible repair operators;

- iv. compute the feasible insertion positions of o according to the active AND/OR precedence relations;
 - v. for each feasible position and each admissible mode:
 - temporarily insert o ;
 - temporarily complete the sequence if required;
 - decode the temporary solution;
 - evaluate the temporary solution.
 - vi. select the position and execution mode giving the best score according to the selected repair operator R ;
 - vii. insert o in π_{repaired} ;
 - viii. update RepairUsage .
- (h) Build the neighbouring solution:

$$S' = (\pi', \mu', z').$$

After the destruction and repair phases, a neighbouring solution S' is obtained. This solution is not evaluated directly from its sequence only. It must first be decoded into an operational line configuration. The decoder assigns operations to stations, checks the station order, determines the required resources, and computes the station loads. In the stochastic version, it also calculates scenario-dependent loads and recourse values.

Algorithm 1 continued: Phase 3 – Decoding and evaluation

- (i) Decode S' :
 - assign operations to stations;
 - compute station loads;
 - compute recourse values in the stochastic case.
- (j) Evaluate S' .
In the deterministic case:

$$F_{\text{det}} = C_{\text{stations}} + C_{\text{execution}} + \text{Penalty}.$$

In the stochastic case:

$$F_{\text{sto}} = C_{\text{stations}} + C_{\text{execution}} + C_{\text{recours}} + \text{Penalty},$$

with:

$$C_{\text{recours}} = \sum_{s \in S} \omega_s \sum_{k \in K} \rho_k r_k^s, \quad r_k^s = \max(0, L_k^s - CT).$$

The fourth phase decides whether the neighbouring solution should replace the current solution. If the new solution improves the objective value, it is directly accepted. If it is worse, it can still be accepted according to a simulated annealing probability. This mechanism helps the search escape local optima by occasionally accepting non-improving moves, especially at the beginning of the search when the temperature is still high.

Algorithm 1 continued: Phase 4 – Acceptance or rejection

(k) Compute:

$$\Delta = F(S') - F(S_{current}).$$

(l) If $\Delta \leq 0$, accept S' :

$$S_{current} \leftarrow S', \quad reward = reward_{medium}.$$

(m) Otherwise, accept S' with simulated annealing probability:

$$p = \exp\left(-\frac{\Delta}{T}\right).$$

If S' is accepted, set:

$$S_{current} \leftarrow S', \quad reward = reward_{low}.$$

Otherwise, reject S' , keep $S_{current}$, and set:

$$reward = 0.$$

The fifth phase updates the best solution found during the search. This step is separated from the acceptance decision because a solution may be accepted as the current solution without necessarily being the best solution obtained so far. Whenever the objective value of the neighbouring solution is better than the best recorded value, the algorithm stores it as the new best solution and resets the stagnation counter.

Algorithm 1 continued: Phase 5 – Best-solution update

(n) If $F(S') < F_{best}$, update the best solution:

$$S^* \leftarrow S', \quad F_{best} \leftarrow F(S'),$$

$$last_improvement_iter \leftarrow iter, \quad reward = reward_{high}.$$

The last phase updates the adaptive components of the algorithm. The destruction and repair operators used during the iteration receive a reward depending on the quality of the solution they helped produce. Operators that contribute to better solutions become more likely to be selected in future iterations. The temperature is also decreased according to the cooling rate, which progressively reduces the probability of accepting worse solutions.

Algorithm 1 continued: Phase 6 – Adaptive update and stopping

(o) Update the weights of the destruction operators used according to their contribution and the value of *reward*.

(p) Update the weights of the repair operators used according to their contribution and the value of *reward*.

(q) Update the temperature:

$$T \leftarrow \max(T_{min}, \beta T).$$

6. Return S^* .

End

Overall, the proposed procedure combines three important mechanisms. The destruction phase diversifies the search by removing selected operations from the current solution. The repair phase intensifies the search by reinserting these operations using problem-specific rules. Finally, the adaptive update and simulated annealing acceptance rule help balance exploration and exploitation. This structure allows the proposed HRC-ALNS to remain close to the general ALNS framework while being adapted to the deterministic and stochastic characteristics of the studied human–robot collaborative disassembly problem.

Table 3.7 summarizes the main differences between the deterministic and stochastic versions of the proposed method.

Table 3.7: Main differences between D-HRC-ALNS and S-HRC-ALNS.

Element	D-HRC-ALNS	S-HRC-ALNS
Processing times	Fixed operation times	Scenario-dependent human operation times
Station loads	Nominal loads	Loads computed for each scenario
Destruction guidance	Bottlenecks, long tasks, costly tasks, normal tasks	Recourse, human variability, worst scenarios, overload risk
Repair guidance	Cost, duration, load balance, station opening	Expected recourse, robustness, scenario load balance, variability separation
Evaluation	Station cost, execution cost and penalty	Station cost, execution cost, expected recourse and penalty
Main objective	Efficient line configuration under fixed data	Line configuration less sensitive to human-time uncertainty

3.12 Conclusion

This chapter introduced the proposed HRC-ALNS methodology for solving the multi-type human–robot collaborative battery disassembly problem. The method is based on the general principle of Adaptive Large Neighbourhood Search, but it is adapted to the specific characteristics of the studied problem. The contribution does not consist in proposing a completely new metaheuristic, but in adapting ALNS to a problem involving several battery types, human/robot/collaborative execution modes, AND/OR precedence relations, station assignment decisions and stochastic human operation times.

The chapter first clarified the methodological positioning of the approach. Exact MILP models remain important because they provide a rigorous formulation and reference solutions for smaller instances. However, their computational cost increases rapidly when the number of battery types, tasks and scenarios grows. This motivates the use of a metaheuristic search procedure capable of exploring larger solution spaces more efficiently.

The proposed encoding represents a solution as $S = (\pi, \mu, z)$, where π is the global operation sequence, μ contains the execution modes, and z stores the active OR choices. Stations are not directly encoded. They are reconstructed by a decoder, which assigns operations to stations, determines resource configurations, computes loads and evaluates recourse in the stochastic case.

The chapter also described the mixed destruction and hybrid repair mechanisms. In the deterministic version, the operators focus on nominal bottlenecks, long tasks, costly tasks and flexible normal tasks. In the stochastic version, the operators additionally exploit scenario information, including recourse, human-time variability and worst-scenario loads. This distinction leads to two variants: D-HRC-ALNS for the deterministic case and S-HRC-ALNS for the stochastic case.

Finally, the chapter presented the evaluation functions, simulated annealing acceptance rule, adaptive operator weight update and general algorithmic framework. The next chapter will use this methodology to conduct computational experiments and compare the obtained metaheuristic solutions with the exact MILP results when available.

Computational Results of the Proposed HRC-ALNS Methodology

4.1 Introduction

The previous chapter introduced the proposed HRC-ALNS methodology developed for the multi-type human–robot collaborative battery disassembly problem. The method was designed as an adaptation of Adaptive Large Neighbourhood Search to the specific structure of the studied problem, including multi-battery task sets, execution mode selection, AND/OR precedence relations, station decoding, and deterministic or stochastic evaluation.

The objective of this chapter is to evaluate the computational performance of the proposed approach. Two versions are considered. The first one is the deterministic version, denoted D-HRC-ALNS, which uses fixed operation times. The second one is the stochastic version, denoted S-HRC-ALNS, which considers scenario-dependent human operation times and includes recourse costs when station loads exceed the common cycle time.

The computational study is organized around three main purposes. First, the proposed method is tested on small instances for which exact MILP results are available. This makes it possible to evaluate the quality of the metaheuristic solutions by comparison with exact reference values. Second, the method is applied to larger instances involving four and five battery types, where exact optimization becomes computationally difficult. Third, the results are analysed in terms of objective value, opened stations, execution cost, recourse cost, CPU time, and optimality gap.

The remainder of this chapter is structured as follows. Section 4.2 presents the experimental design. Section 4.3 compares the proposed HRC-ALNS with the exact MILP results. Sections 4.4 and 4.5 present the deterministic and stochastic results, respectively. Section 4.6 discusses the gap and CPU time comparison. Section 4.7 provides a graphical interpretation of the obtained schedules. Finally, the chapter ends with a general discussion and a conclusion.

4.2 Experimental Design

This section presents the experimental setting used to evaluate the proposed HRC-ALNS methodology. The same general structure is used for the deterministic and stochastic versions. The difference between the two cases concerns the treatment of operation times and recourse. In the deterministic case, operation times are fixed. In the stochastic case, human operation times depend on scenarios, while robot and collaborative operation times remain deterministic.

4.2.1 Implementation Environment

The proposed HRC-ALNS algorithm was implemented in Python 3. The experiments were carried out on a MacBook Pro Retina 13-inch, Early 2015, running macOS 11.6.7. The implementation uses Python libraries for data handling, Excel export and graphical representation of the obtained schedules.

The algorithm generates both numerical results and visual outputs. The numerical results are exported to Excel files, while the schedules are represented through Gantt charts. These graphical outputs are used only for interpretation and do not modify the optimization results.

4.2.2 Test Instances

The proposed method was tested on instances involving one to five battery types. The one-, two- and three-battery instances correspond to the exact MILP validation cases previously used as reference. The four- and five-battery instances extend the computational study in order to evaluate the scalability of the proposed HRC-ALNS methodology when the exact MILP formulation becomes computationally restrictive.

The composition of the tested instances is summarized in Table 4.1.

Table 4.1: Composition of the tested HRC-ALNS instances

Instance	Battery types included	Total tasks
1 battery	Tesla Model 1 [58]	22
2 batteries	Tesla Model 1 [58] and Tesla Model S [12]	44
3 batteries	Tesla Model 1 [58], Tesla Model S [12], and Guangqi Toyota LB7A-FX1 [50]	68
4 batteries	Tesla Model 1 [58], Tesla Model S [12], Guangqi Toyota LB7A-FX1 [50], and Red Flag [31]	94
5 batteries	Tesla Model 1 [58], Tesla Model S [12], Guangqi Toyota LB7A-FX1 [50], Red Flag [31], and Tesla module with 44 tasks [49]	138

Each instance was solved in both deterministic and stochastic configurations. The deterministic version is denoted D-HRC-ALNS, while the stochastic version is denoted S-HRC-ALNS. The one-, two- and three-battery instances are used to validate the proposed method against exact MILP results. The four- and five-battery instances are used only to analyse the scalability of the metaheuristic, since no exact MILP reference solution is available for these larger cases.

4.2.3 Main Parameter Settings

The main cost and configuration parameters are kept consistent with the exact models used as reference. The station opening cost is fixed at 5000. The human unit cost is equal to 0.15, while the robot-related cost is calculated using the robot cost coefficient and the energy coefficient. In the stochastic case, the recourse cost is fixed at 200 per second of overload. The allowed recourse is bounded by 15% of the cycle time.

The common cycle time is obtained from the stochastic reference duration and is used in both deterministic and stochastic experiments. This choice ensures that both versions are compared under the same line-capacity reference.

Table 4.2: Main HRC-ALNS parameter settings

Parameter	Value
Station opening cost	5000
Human unit cost	0.15
Robot cost coefficient	0.1575
Energy coefficient	0.275
Cycle-time factor	2.0
Recourse cost per second	200
Maximum recourse ratio	0.15
Minimum destruction ratio	0.10
Maximum destruction ratio	0.35
Maximum iterations per restart	800
Cooling rate	0.997
Minimum temperature	10^{-6}

At each iteration, the number of removed operations is randomly selected between 10% and 35% of the total number of operations. This variable destruction size allows the algorithm to alternate between small local modifications and larger perturbations. Small removals support intensification around promising solutions, while larger removals help diversify the search.

4.2.4 Stopping Criteria

The stopping criterion is based on the stagnation of the best solution. The algorithm stops when no improvement of the best objective value is observed during a predefined number of consecutive iterations.

The deterministic and stochastic versions use different stagnation limits. In the deterministic case, the algorithm stops after 250 consecutive iterations without improvement. In the stochastic case, the limit is increased to 500 consecutive iterations without improvement because each solution evaluation requires the computation of scenario-dependent station loads and recourse costs.

Table 4.3: Stopping criteria used in the HRC-ALNS experiments

Version	Initial temperature	Stagnation limit
D-HRC-ALNS	250	250 iterations without improvement
S-HRC-ALNS	500	500 iterations without improvement

During the search, the temperature is progressively reduced according to the simulated annealing cooling rule. This mechanism allows the algorithm to accept some non-improving solutions at the beginning of the search. As the temperature decreases, the probability of accepting worse solutions becomes lower, and the search gradually focuses on improving the best solution found.

4.2.5 Performance Indicators

Several indicators are used to evaluate the proposed method. The first indicator is the objective value obtained by HRC-ALNS. This value includes the station opening cost, the execution cost, the recourse cost in the stochastic case, and possible penalties for infeasible solutions. The second indicator is the number of opened stations. The third indicator is the CPU time required by the algorithm.

For instances where exact MILP results are available, the quality of the metaheuristic solution is evaluated using the relative gap:

$$Gap(\%) = \frac{F_{HRC-ALNS} - F_{MILP}}{F_{MILP}} \times 100. \quad (4.1)$$

A gap equal to zero means that the metaheuristic reaches the same objective value as the exact MILP. A positive gap indicates that the HRC-ALNS solution is more expensive than the exact reference solution. For larger instances, where the exact model becomes difficult to solve, the analysis focuses on the ability of HRC-ALNS to generate feasible and interpretable solutions within a reasonable computational time.

4.3 Validation of HRC-ALNS Against Exact MILP Results

Before applying the proposed HRC-ALNS methodology to larger instances, the method is first validated on small instances for which exact MILP results are available. This validation step is important because it makes it possible to assess whether the metaheuristic is able to reproduce, or at least closely approximate, the solutions obtained by the exact mathematical formulation.

The comparison is performed for the one-, two- and three-battery instances, in both deterministic and stochastic cases. For each instance, the objective value obtained by HRC-ALNS is compared with the objective value obtained by the exact MILP. The relative gap is calculated using the formula presented in Section 4.2.

In addition to the objective value, two CPU time indicators are reported for the metaheuristic. The first one is the time required to reach the best solution found during the search. The second one is the total execution time until the stopping criterion is reached. This distinction is important because the HRC-ALNS may find its best solution before the end of the search, while the remaining iterations are used to confirm the absence of further improvement.

4.3.1 Deterministic Validation

Table 4.4 presents the comparison between the deterministic MILP and D-HRC-ALNS for the one-, two- and three-battery instances.

Table 4.4: Validation of D-HRC-ALNS against deterministic MILP results

Instance	D-HRC-ALNS objective	Time to best (s)	Total time (s)	MILP objective	Gap (%)
1 battery	5046.4869	0.50	2.42	5046.4869	0.000000
2 batteries	15087.4450	3.70	18.80	15087.0287	0.002759
3 batteries	29971.6941	16.29	67.48	29971.6941	0.000001

The deterministic results show that D-HRC-ALNS provides solutions that are very close to the exact MILP solutions. For the one-battery instance, the metaheuristic reaches exactly the same objective value as the MILP. For the two-battery instance, the relative gap is only 0.002759%, which is negligible in practice. For the three-battery instance, the obtained objective value is again equivalent to the MILP reference value, with a gap close to zero.

The CPU time comparison also highlights the computational interest of the proposed metaheuristic. For the three-battery instance, the exact MILP requires 557.9806 seconds, while D-HRC-ALNS finds its best solution after only 16.29 seconds and stops after 67.48 seconds. This confirms that the metaheuristic is able to reach high-quality solutions much faster than the exact formulation on the deterministic validation instances.

4.3.2 Stochastic Validation

Table 4.5 presents the comparison between the stochastic MILP and S-HRC-ALNS for the same set of instances.

Table 4.5: Validation of S-HRC-ALNS against stochastic MILP results

Instance	S-HRC-ALNS objective	Time to best (s)	Total time (s)	MILP objective	Gap (%)
1 battery	5460.5824	2.80	23.18	5460.5824	0.000000
2 batteries	15088.6599	86.70	174.20	15087.0287	0.010810
3 batteries	29978.0171	470.92	811.60	29971.6941	0.021092

The stochastic validation results confirm the same general trend. For the one-battery instance, S-HRC-ALNS reaches the same objective value as the exact stochastic MILP. For the two- and three-battery instances, the gaps remain very small, equal to 0.010810% and 0.021092%, respectively. These values indicate that the stochastic version of the proposed metaheuristic remains very close to the exact reference solutions, despite the additional complexity introduced by scenario-dependent human operation times and recourse costs.

The CPU time comparison is particularly significant in the stochastic case. For the three-battery instance, the stochastic MILP requires 15092.8327 seconds, whereas S-HRC-ALNS finds its best solution after 470.92 seconds and completes the search after 811.60 seconds. This shows that the proposed method considerably reduces the computational effort while maintaining a very small optimality gap.

Overall, the validation results show that the proposed HRC-ALNS methodology is able to reproduce the exact MILP solutions or generate very close approximations on small instances. This provides confidence in the quality of the metaheuristic before applying it to larger instances where exact optimization is no longer available.

4.4 Deterministic HRC-ALNS Results

This section presents the results obtained with the deterministic version of the proposed metaheuristic, denoted D-HRC-ALNS. In this case, operation times are fixed and the objective function includes the station opening cost, the execution cost and possible penalties. Since the deterministic evaluation does not include scenarios, no recourse cost is considered.

The deterministic experiments were carried out on the five tested instances, from one battery to five batteries. For the one-, two- and three-battery instances, exact MILP results

are available and were already used for validation in the previous section. For the four- and five-battery instances, no exact MILP result is available. These two cases are therefore used to evaluate the ability of the proposed metaheuristic to generate feasible solutions for larger instances.

Table 4.6 summarizes the deterministic results obtained with D-HRC-ALNS.

Table 4.6: Deterministic results obtained with D-HRC-ALNS

Instance	Objective	Station cost	Execution cost	Opened stations	Time to best (s)	Total time (s)
1 battery	5046.4869	5000	46.4869	1	0.50	2.42
2 batteries	15087.4450	15000	87.4450	3	3.70	18.80
3 batteries	29971.6941	25000	4971.6941	5	16.29	67.48
4 batteries	18967.7876	15000	3967.7876	3	135.39	308.97
5 batteries	23229.2595	20000	3229.2595	4	405.30	937.10

The results show that D-HRC-ALNS is able to generate feasible deterministic solutions for all tested instances. In all cases, the penalty value is equal to zero, which confirms that the solutions respect the main feasibility requirements imposed by the decoder, including precedence relations, station order and cycle-time limitations.

For the one-battery instance, the algorithm opens one station and reaches the best solution in only 0.50 seconds. This result confirms that the proposed encoding and decoding mechanism is able to reproduce the exact solution structure for a small instance. For the two-battery instance, the number of opened stations increases to three, with a total objective value of 15087.4450. The best solution is reached after 3.70 seconds, while the complete search stops after 18.80 seconds.

For the three-battery instance, D-HRC-ALNS opens five stations and obtains an objective value of 29971.6941. Although the instance is larger, the best solution is found after 16.29 seconds, and the total execution time remains limited to 67.48 seconds. This confirms that the deterministic version of the proposed method remains computationally efficient for instances where the exact MILP already requires a much longer resolution time.

For the four- and five-battery instances, the absence of MILP reference values prevents the calculation of an optimality gap. However, the obtained results show that the proposed metaheuristic remains able to construct feasible and interpretable line configurations. The four-battery instance is solved with three opened stations and a total execution time of 308.97 seconds. The five-battery instance requires four opened stations and a total execution time of 937.10 seconds.

It is important to note that the objective values should not be interpreted only as a function of the number of batteries. The instances differ in terms of product composition, task durations, execution modes and weighted execution costs. Therefore, the increase in the number of battery types does not necessarily lead to a strictly increasing objective value. The most relevant interpretation is related to feasibility, number of opened stations, computational time and scalability.

Overall, the deterministic results show that D-HRC-ALNS provides high-quality solutions for small instances and remains applicable to larger instances where exact optimization is not available. The distinction between the time to best solution and the total execution time also shows that the algorithm often identifies its best solution before the stopping criterion is reached. The remaining iterations are mainly used to verify that no further improvement can be obtained within the allowed stagnation limit.

4.5 Stochastic HRC-ALNS Results

This section presents the results obtained with the stochastic version of the proposed metaheuristic, denoted S-HRC-ALNS. In this version, human operation times are scenario-dependent, while robot and collaborative operation times remain deterministic. The objective function includes station opening cost, weighted execution cost, expected recourse cost and possible penalties.

The stochastic experiments were carried out on the five tested instances. For the one-, two- and three-battery instances, exact MILP results were available and were used in the validation section. For the four- and five-battery instances, no MILP result was available, and the analysis therefore focuses on the ability of S-HRC-ALNS to generate feasible solutions under uncertainty.

Table 4.7 summarizes the stochastic results obtained with S-HRC-ALNS.

Table 4.7: Stochastic results obtained with S-HRC-ALNS

Instance	Objective	Station cost	Execution cost	Expected recourse	Opened stations	Time to best (s)	Total time (s)
1 battery	5460.5824	5000	48.6224	411.9600	1	2.80	23.18
2 batteries	15088.6599	15000	88.6599	0.0000	3	86.73	174.20
3 batteries	29978.0171	25000	4978.0171	0.0000	5	470.92	811.60
4 batteries	18968.9719	15000	3968.9719	0.0000	3	1630.09	2656.63
5 batteries	23232.9465	20000	3232.9465	0.0000	4	5519.40	8395.21

The results show that S-HRC-ALNS is able to generate feasible stochastic solutions for all tested instances. In all cases, the penalty value is equal to zero, which means that the solutions returned by the decoder satisfy the main feasibility requirements, including precedence relations, station order, resource compatibility and the allowed recourse bound.

For the one-battery instance, the algorithm opens one station and obtains an objective value of 5460.5824. In this case, an expected recourse cost of 411.9600 is observed. This means that, under some human-time scenarios, the station load exceeds the common cycle time, but the overload remains within the authorized recourse limit. Therefore, the solution remains feasible, but the uncertainty generates an additional cost.

For the two- and three-battery instances, the expected recourse cost is equal to zero. This indicates that the selected station configurations are robust enough to absorb the scenario-dependent variation of human operation times without exceeding the common cycle time. The two-battery instance opens three stations, while the three-battery instance opens five stations. These results are consistent with the increase in the number of operations and precedence constraints.

For the four- and five-battery instances, no exact MILP solution is available. Nevertheless, S-HRC-ALNS provides feasible line configurations with zero penalty and zero expected recourse cost. The four-battery instance is solved with three opened stations and an objective value of 18968.9719. The five-battery instance is solved with four opened stations and an objective value of 23232.9465.

The CPU time increases with the size and complexity of the instance. This behaviour is expected because the stochastic version evaluates each candidate solution across all scenarios. For the five-battery instance, the best solution is reached after 5519.40 seconds, while the total execution time until stagnation is 8395.21 seconds. This distinction shows that the best solution is found before the end of the search, and that the remaining iterations mainly confirm that no better solution is found within the stagnation limit.

Overall, the stochastic results confirm that S-HRC-ALNS remains applicable even when uncertainty is introduced. The method is able to combine sequence construction,

execution mode selection, station decoding and recourse evaluation within the same search process. The absence of penalties in all instances also confirms the reliability of the decoding and repair mechanisms used to maintain feasible solutions.

4.6 Comparison and Gap Analysis

This section compares the computational behaviour of the exact MILP approach and the proposed HRC-ALNS methodology. The comparison is based on objective values, optimality gaps and CPU times. For the one-, two- and three-battery instances, MILP results are available and can therefore be used as reference values. For the four- and five-battery instances, the MILP results are not available, which confirms the computational difficulty of the exact formulation for larger cases.

4.6.1 Deterministic Comparison

Table 4.8 summarizes the deterministic comparison between the exact MILP and D-HRC-ALNS.

Table 4.8: Deterministic comparison between MILP and D-HRC-ALNS

Instance	D-HRC-ALNS objective	Time to best (s)	Total time (s)	MILP objective	MILP CPU time (s)	Gap (%)
1 battery	5046.4869	0.50	2.42	5046.4869	2.3920	0.000000
2 batteries	15087.4450	3.70	18.80	15087.0287	35.4773	0.002759
3 batteries	29971.6941	16.29	67.48	29971.6941	557.9806	0.000001
4 batteries	18967.7876	135.39	308.97	–	–	–
5 batteries	23229.2595	405.30	937.10	–	–	–

For the deterministic case, the results show that D-HRC-ALNS reaches solutions that are almost identical to the exact MILP results on the validation instances. The gap is equal to zero for the one-battery instance, very close to zero for the three-battery instance, and only 0.002759% for the two-battery instance. These values confirm that the proposed metaheuristic preserves a very high level of solution quality.

The CPU time comparison shows the computational advantage of D-HRC-ALNS. For the two-battery instance, the MILP requires 35.4773 seconds, while the metaheuristic finds its best solution in 3.70 seconds and stops after 18.80 seconds. For the three-battery instance, the difference becomes more significant. The MILP requires 557.9806 seconds, whereas D-HRC-ALNS reaches its best solution in 16.29 seconds and stops after 67.48 seconds.

For the four- and five-battery deterministic instances, no MILP reference is available. In these cases, D-HRC-ALNS is used as the main resolution approach. The method provides feasible solutions within 308.97 seconds for the four-battery instance and 937.10 seconds for the five-battery instance. These results show that the proposed method remains applicable when the exact formulation becomes difficult to use.

4.6.2 Stochastic Comparison

Table 4.9 presents the stochastic comparison between the exact MILP and S-HRC-ALNS.

Table 4.9: Stochastic comparison between MILP and S-HRC-ALNS

Instance	S-HRC-ALNS objective	Time to best (s)	Total time (s)	MILP objective	MILP CPU time (s)	Gap (%)
1 battery	5460.5824	2.80	23.18	5460.5824	42.5012	0.000000
2 batteries	15088.6599	86.70	174.20	15087.0287	698.8258	0.010810
3 batteries	29978.0171	470.92	811.60	29971.6941	15092.8327	0.021092
4 batteries	18968.9719	1630.09	2656.63	–	–	–
5 batteries	23232.9465	5519.40	8395.21	–	–	–

The stochastic comparison confirms the same general behaviour, but with higher CPU times due to the additional scenario-based evaluation. For the one-battery instance, S-HRC-ALNS reaches exactly the same objective value as the stochastic MILP. For the two- and three-battery instances, the gaps remain very small, equal to 0.010810% and 0.021092%, respectively.

The CPU time reduction is particularly important in the stochastic case. For the three-battery instance, the exact MILP requires 15092.8327 seconds, while S-HRC-ALNS finds its best solution after 470.92 seconds and stops after 811.60 seconds. This result shows that the proposed stochastic metaheuristic provides near-optimal solutions with a much lower computational effort.

For the four- and five-battery stochastic instances, the MILP results are not available. Nevertheless, S-HRC-ALNS produces feasible solutions while explicitly considering scenario-dependent human operation times and recourse costs. The increase in total CPU time for these instances is expected because each candidate solution must be decoded and evaluated across all scenarios.

4.6.3 Interpretation of the Gap Results

The gap values obtained for the validation instances are very small in both deterministic and stochastic cases. This confirms that the proposed HRC-ALNS methodology is able to reproduce the behaviour of the exact model on small instances while requiring less computational time.

The deterministic gaps are almost equal to zero, which indicates that the deterministic version of the metaheuristic is very effective when operation times are fixed. The stochastic gaps are slightly higher, but they remain below 0.03%. This increase is expected because the stochastic case is more complex: the same solution must remain efficient over several scenarios, and the evaluation includes expected recourse.

The comparison also shows that the time required to reach the best solution is always smaller than the total execution time. This means that the algorithm often identifies its best solution before the stopping criterion is reached. The remaining iterations are used to continue the exploration and confirm that no better solution is found within the stagnation limit.

Overall, the comparison demonstrates that HRC-ALNS offers a strong compromise between solution quality and computational effort. It provides solutions that are very close to the MILP results for small instances and remains applicable to larger instances where the exact model is no longer available.

4.7 Graphical Analysis of the Obtained Schedules

In addition to the numerical results, the obtained solutions are analysed graphically using Gantt charts. These diagrams make it possible to visualize the assignment of tasks to

stations, the use of human and robotic resources, the position of collaborative tasks, and the compacted execution time inside each station.

The Gantt charts are not used to compute the objective value. They are generated after the optimization process in order to interpret the final decoded solution. Each opened station is represented with its corresponding resources. When a station contains both human and robotic resources, the station is divided into two lines: one for the human operator and one for the robot. Collaborative tasks appear on both lines because they require simultaneous human–robot execution.

4.7.1 Deterministic Gantt Charts

The deterministic Gantt charts represent the schedules obtained with D-HRC-ALNS using nominal operation times. These charts allow the verification of the compacted workload inside each station and show how the decoder distributes human, robotic and collaborative tasks.

Figures 4.1–4.5 present the deterministic Gantt charts for the tested instances.

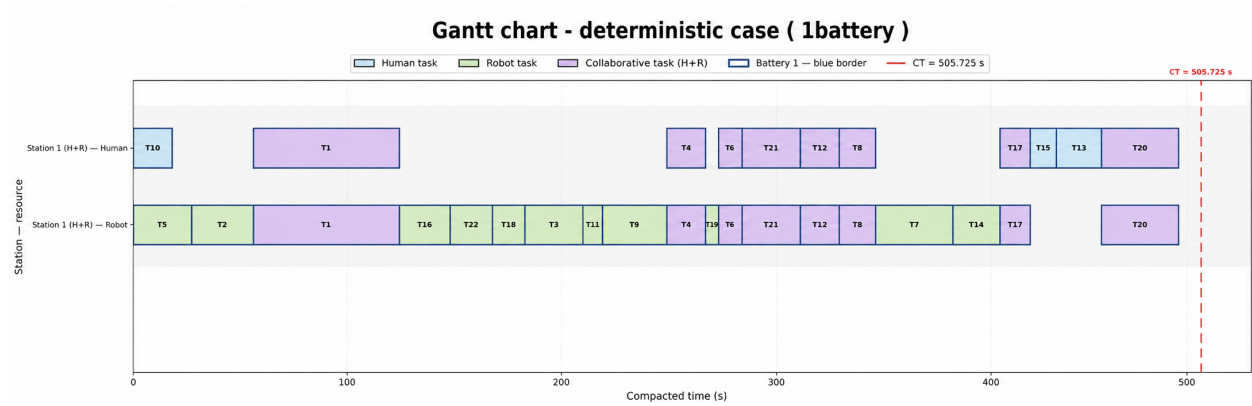


Figure 4.1: Deterministic Gantt chart obtained with D-HRC-ALNS for the one-battery instance.

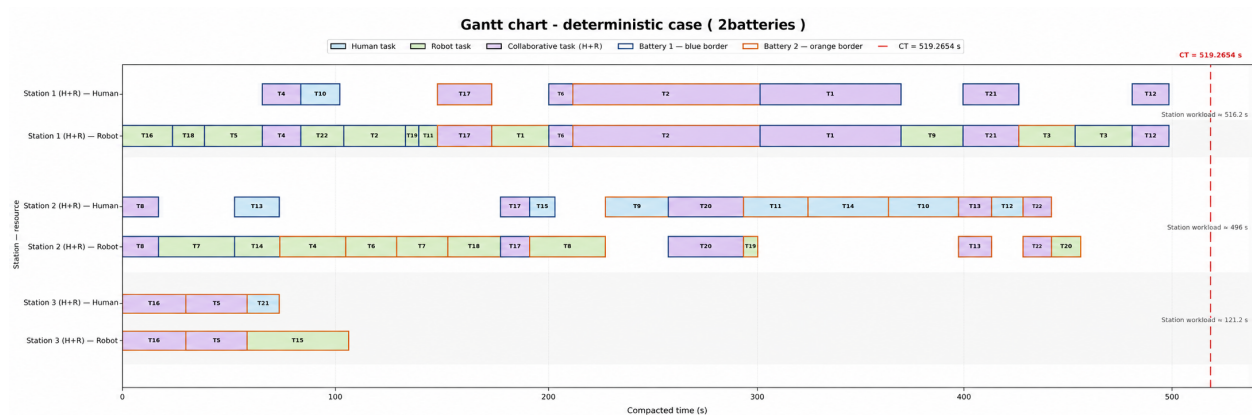


Figure 4.2: Deterministic Gantt chart obtained with D-HRC-ALNS for the two-battery instance.

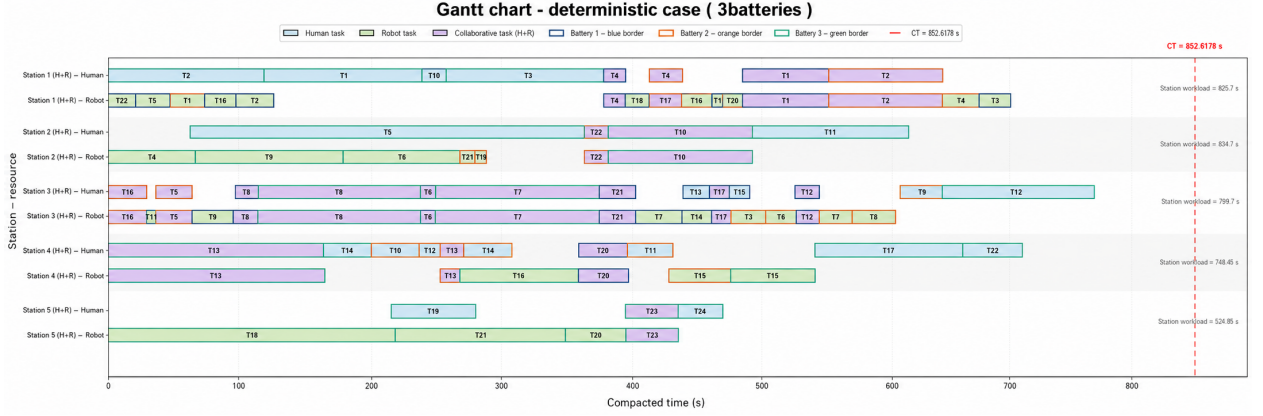


Figure 4.3: Deterministic Gantt chart obtained with D-HRC-ALNS for the three-battery instance.

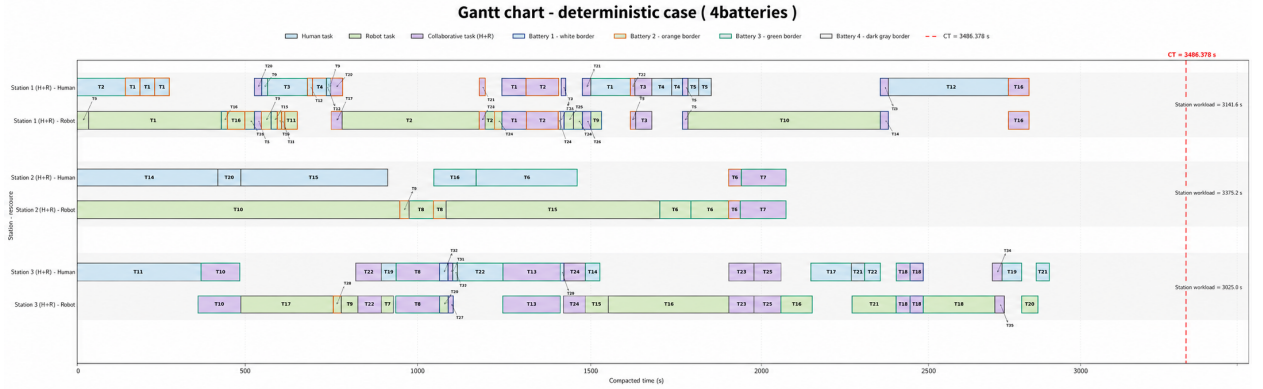


Figure 4.4: Deterministic Gantt chart obtained with D-HRC-ALNS for the four-battery instance.

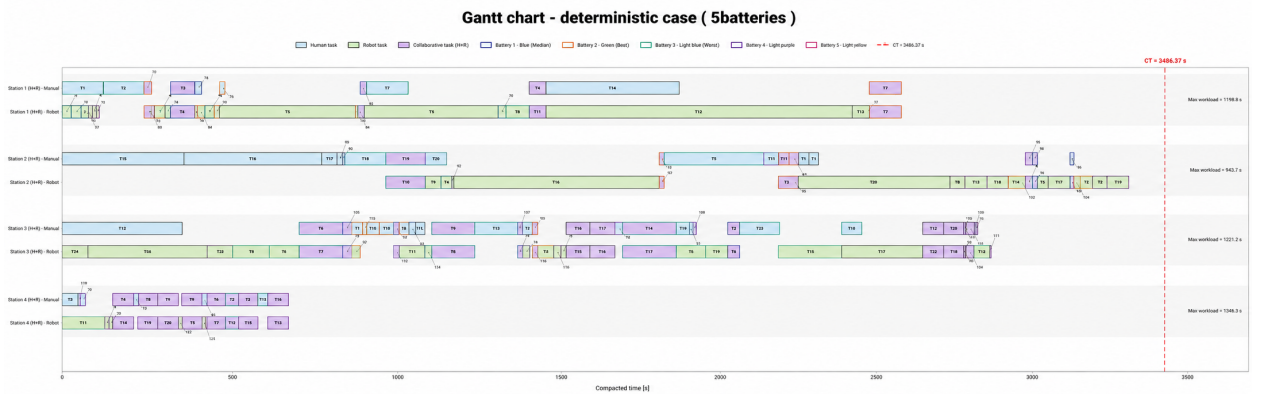


Figure 4.5: Deterministic Gantt chart obtained with D-HRC-ALNS for the five-battery instance.

The deterministic Gantt charts show that the decoder produces compact line configurations while respecting the physical order of stations. Human-only tasks, robot-only tasks and collaborative tasks are assigned to compatible stations. The presence of collaborative tasks on both human and robot lines confirms that these operations require both resources at the same station.

For the larger instances, the charts are particularly useful because the numerical objective value alone does not show how the tasks are distributed. The visual representation helps identify station workloads, possible bottleneck stations and the distribution of tasks among resources.

4.7.2 Stochastic Gantt Charts

The stochastic Gantt charts are generated for scenario 1 in order to illustrate one possible realization of the uncertain human operation times. The line configuration itself remains the solution obtained by S-HRC-ALNS, while the displayed task durations correspond to the selected scenario.

Figures 4.6–4.10 present the stochastic Gantt charts for the tested instances.

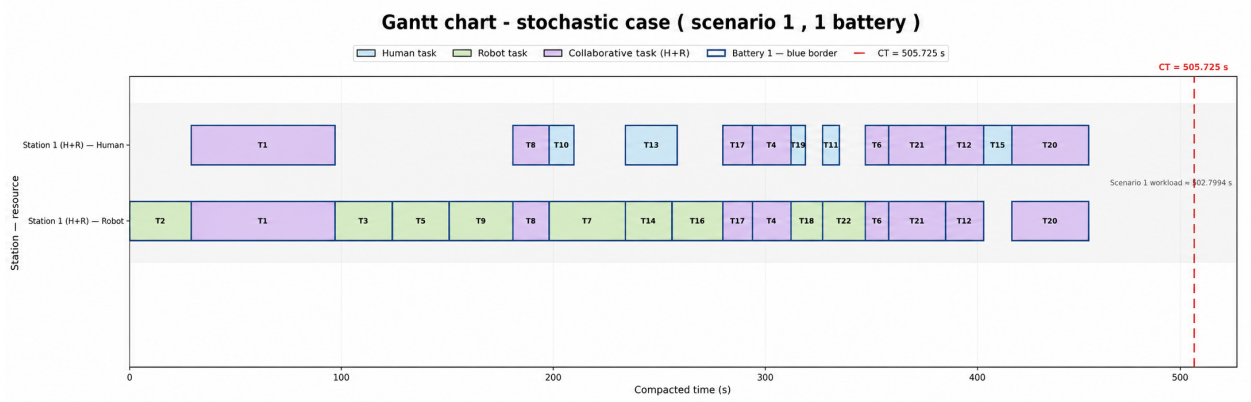


Figure 4.6: Stochastic Gantt chart obtained with S-HRC-ALNS for the one-battery instance, scenario 1.

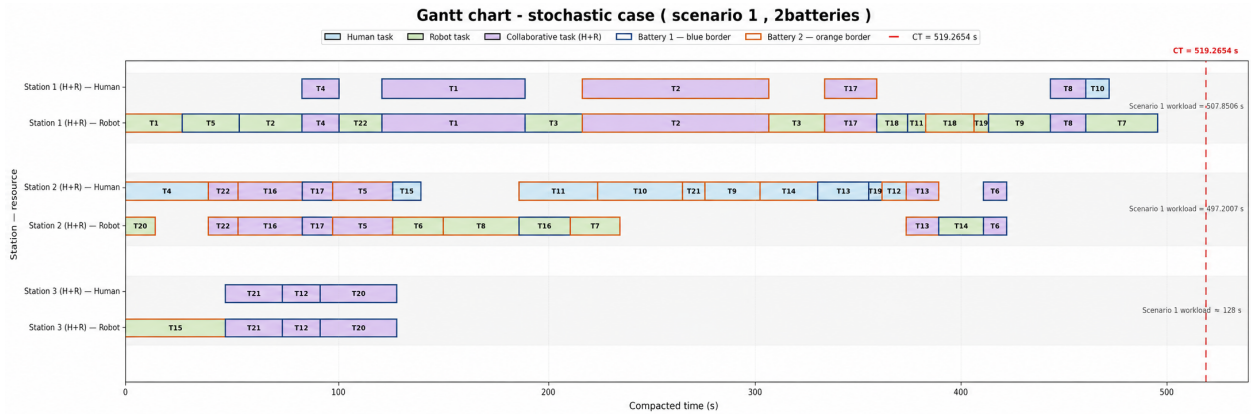


Figure 4.7: Stochastic Gantt chart obtained with S-HRC-ALNS for the two-battery instance, scenario 1.

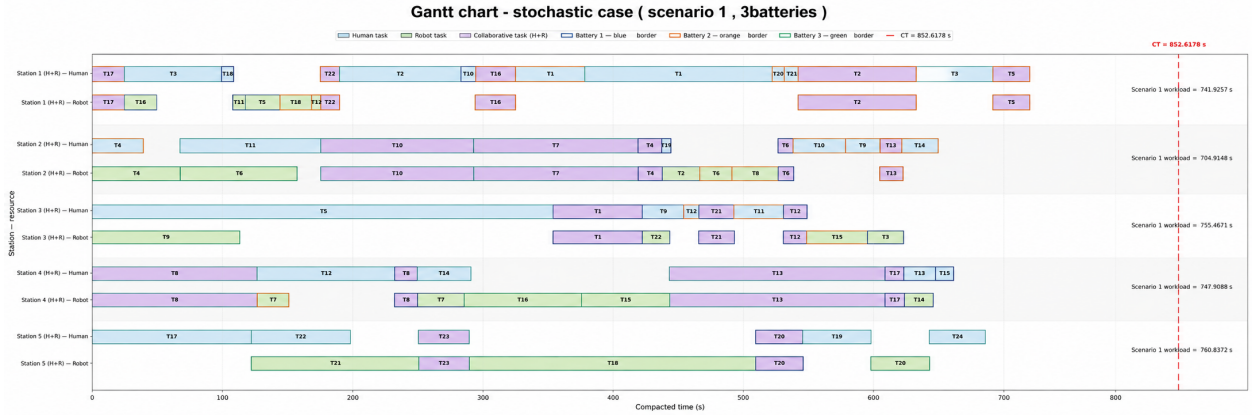


Figure 4.8: Stochastic Gantt chart obtained with S-HRC-ALNS for the three-battery instance, scenario 1.

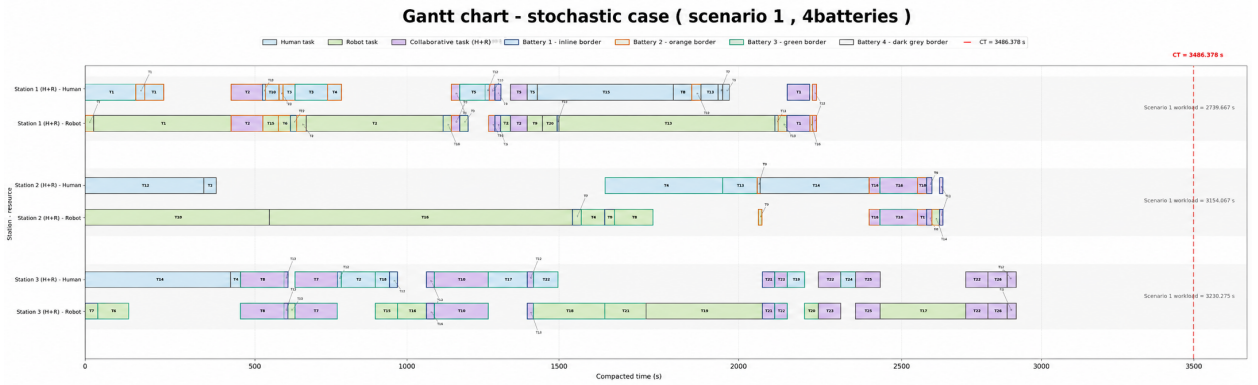


Figure 4.9: Stochastic Gantt chart obtained with S-HRC-ALNS for the four-battery instance, scenario 1.

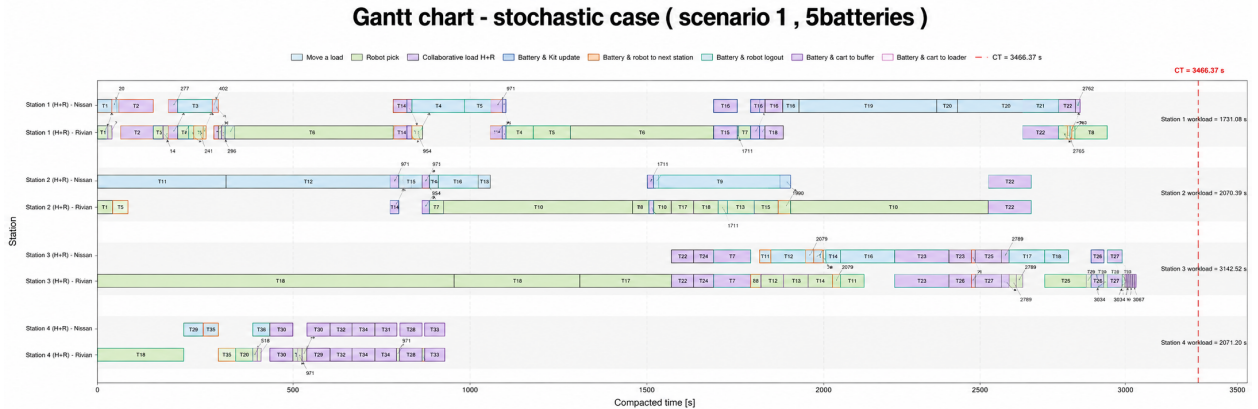


Figure 4.10: Stochastic Gantt chart obtained with S-HRC-ALNS for the five-battery instance, scenario 1.

The stochastic Gantt charts provide an interpretation of how uncertainty affects the final schedule. Human task durations may vary from one scenario to another, which can modify station loads and create recourse in some cases. However, the obtained solutions remain feasible because the recourse is bounded and penalties are avoided.

For the one-battery stochastic instance, the numerical results show a positive expected recourse cost. This means that some scenario-dependent loads exceed the common cycle time within the allowed limit. For the other stochastic instances, the expected recourse cost is equal to zero, which indicates that the station configurations obtained by S-HRC-ALNS are sufficiently robust with respect to the tested scenarios.

Overall, the graphical analysis supports the numerical results. It confirms that the proposed decoder generates coherent station assignments, that the resource compatibility constraints are respected, and that collaborative tasks are correctly represented on both human and robot resource lines.

4.8 Discussion

The computational results provide several observations regarding the behaviour of the proposed HRC-ALNS methodology. The first observation concerns solution quality. On the instances where exact MILP results are available, the obtained gaps are extremely small. This confirms that the proposed metaheuristic is able to approximate the exact mathematical model very closely. In the deterministic case, the gap is almost zero for all validation instances. In the stochastic case, the gaps remain below 0.03%, despite the additional complexity introduced by scenario-dependent human operation times and recourse evaluation.

The second observation concerns computational time. The exact MILP is useful for small instances because it provides reference optimal solutions, but its computational cost increases rapidly as the number of batteries and tasks grows. In contrast, HRC-ALNS is able to reach high-quality solutions in shorter times for the validation instances and remains applicable to the four- and five-battery instances. This confirms the interest of using the metaheuristic as a complementary approach when the exact formulation becomes computationally restrictive.

The third observation concerns the difference between the deterministic and stochastic versions. S-HRC-ALNS requires higher CPU times than D-HRC-ALNS because each candidate solution must be evaluated over all scenarios. This additional computational effort is expected. However, the stochastic version provides more information because it evaluates the robustness of station loads under uncertain human operation times. The presence or absence of recourse costs allows a better understanding of the sensitivity of the line configuration to uncertainty.

The fourth observation concerns the role of the decoder. The results show that the proposed encoding alone is not sufficient; the decoder is essential to transform the internal representation into a feasible line configuration. By assigning tasks to stations after each modification of the sequence and modes, the decoder maintains the physical station order, respects precedence relations and computes station loads consistently. This validates the methodological choice made in Chapter 3, where stations were not directly encoded in the solution representation.

The fifth observation concerns the mixed destruction and hybrid repair mechanisms. The results indicate that these mechanisms are able to modify the solution without destroying its feasibility. By targeting bottleneck stations, long tasks, costly tasks, normal tasks, recourse stations or high-variability human tasks, the destruction phase explores relevant parts of the solution. The repair phase then reinserts removed operations using rules adapted to the cause of their removal. This coupling contributes to the stability of the search and to the quality of the final solutions.

For the large instances, the absence of MILP solutions means that no exact optimality gap can be reported. Therefore, the interpretation focuses on feasibility, computational time and the structure of the obtained schedules. The fact that the proposed method generates feasible solutions with zero penalty for the four- and five-battery instances supports the scalability of the approach. Nevertheless, since the optimal solution is unknown for these instances, the obtained results should be interpreted as high-quality feasible solutions rather than proven optimal solutions.

Overall, the results confirm the relevance of the proposed HRC-ALNS methodology. The method preserves the modelling logic of the exact formulation while providing a more flexible and scalable search process. This makes it suitable for larger multi-battery disassembly instances where exact optimization alone is not sufficient.

4.9 Conclusion

This chapter presented the computational results obtained with the proposed HRC-ALNS methodology. The deterministic and stochastic versions were tested on instances involving one to five battery types. The one-, two- and three-battery instances were used for validation against exact MILP results, while the four- and five-battery instances were used to assess the scalability of the proposed approach.

The validation results show that D-HRC-ALNS and S-HRC-ALNS produce solutions that are very close to the exact MILP solutions. In the deterministic case, the gaps are almost equal to zero. In the stochastic case, the gaps remain very small, even when scenario-dependent human operation times and recourse costs are considered. These results confirm the quality of the proposed metaheuristic on instances where exact reference values are available.

The computational results also show that HRC-ALNS reduces the resolution effort compared with the exact formulation. The distinction between the time to best solution and the total time until stagnation shows that the best solution is often reached before the end of the search. The remaining iterations mainly serve to continue the exploration and confirm that no further improvement is found within the stopping criterion.

For the four- and five-battery instances, the exact MILP results were not available. However, the proposed metaheuristic was able to generate feasible and interpretable solutions in both deterministic and stochastic cases. This confirms the usefulness of HRC-ALNS as a scalable approach for larger multi-type battery disassembly line balancing problems.

Finally, the graphical analysis based on Gantt charts provided a visual interpretation of the obtained solutions. It showed the distribution of tasks across stations, the use of human and robotic resources, and the placement of collaborative operations. These visual results support the numerical analysis and confirm that the final decoded schedules are coherent with the structure of the studied human-robot collaborative disassembly problem.

The next part of the dissertation summarizes the main conclusions of the work, discusses its limitations, and presents possible perspectives for future research.

General Conclusion

This thesis addressed the optimization of a multi-type human–robot collaborative battery disassembly line. The studied problem belongs to the family of complex industrial optimization problems because it combines several interdependent decisions: task sequencing, station opening, task-to-station assignment, execution mode selection, resource compatibility, precedence constraints, cycle-time limitations, and uncertainty in human operation times.

The work was motivated by the fact that exact mathematical models, although rigorous and useful for small instances, become increasingly difficult to solve when the number of battery types, tasks, stations, resources, and scenarios increases. In such a context, a scalable solution approach is required in order to obtain good-quality solutions within reasonable computational time.

The first chapter introduced the main optimization concepts required to understand the methodological background of the thesis. It presented the structure of optimization problems, mathematical programming, line balancing, scheduling, exact methods, heuristics, metaheuristics, and optimization under uncertainty. This chapter showed why industrial planning problems can become highly combinatorial and why exact methods may need to be complemented by metaheuristic approaches for larger instances.

The second chapter reviewed the literature related to disassembly planning, electric vehicle battery disassembly, human–robot collaboration, uncertainty modelling, and multi-product disassembly lines. The review showed that existing studies have addressed several important aspects of the problem, such as disassembly sequence planning, line balancing, robotic disassembly, stochastic task durations, and human–robot collaboration. However, the simultaneous integration of multi-type battery disassembly, human/robot/-collaborative execution modes, AND/OR precedence relations, and stochastic human operation times remains limited.

The third chapter presented the proposed HRC-ALNS methodology. The method was not introduced as a completely new independent metaheuristic, but as a problem-specific adaptation of Adaptive Large Neighbourhood Search to the studied disassembly line problem. The proposed solution representation is based on three components: the global operation sequence, the execution mode vector, and the active OR precedence choices. Stations are not directly encoded; they are reconstructed by a decoder that assigns operations to stations, determines station types, computes station loads, and evaluates the objective function.

The fourth chapter evaluated the proposed methodology through computational experiments. The method was tested on instances involving one to five battery types, in both deterministic and stochastic cases. For the one-, two-, and three-battery instances, exact MILP results were available and used as reference solutions. For the four- and five-battery instances, the MILP reference was not available, and the proposed metaheuristic was used as the main solution approach.

Main Contribution and Added Value

The main contribution of this thesis is the adaptation of an ALNS-based metaheuristic to a multi-type human–robot collaborative battery disassembly line problem. The proposed

method, denoted HRC-ALNS, integrates the main structural characteristics of the studied problem, including several battery types, task precedence relations, execution mode selection, station decoding, and deterministic or stochastic evaluation.

A first added value lies in the proposed solution encoding. Instead of directly encoding station assignments, the solution is represented by the triplet $S = (\pi, \mu, z)$, where π represents the global operation sequence, μ represents the execution modes, and z represents the active OR choices. This representation makes the search more flexible and avoids generating too many infeasible solutions during the destruction and repair phases.

A second added value is the decoding mechanism. The decoder transforms the internal representation into a complete line configuration by assigning operations to stations, respecting the physical order of stations, determining the required resources, computing station loads, and evaluating feasibility. In the stochastic case, the decoder also evaluates scenario-dependent loads and recourse values.

A third contribution concerns the design of mixed destruction and hybrid repair mechanisms. In the deterministic version, the destruction and repair operators target bottleneck stations, long tasks, costly tasks, and flexible normal tasks. In the stochastic version, additional information is used, such as recourse, human-time variability, and worst-scenario station loads. This allows the method to adapt the search process to the real weaknesses of each candidate solution.

The computational results confirmed the relevance of the proposed approach. For the validation instances, the gaps between HRC-ALNS and the exact MILP results remained very small. In the deterministic case, the maximum gap was only 0.002759%. In the stochastic case, the gaps were also limited, reaching 0.010810% for the two-battery instance and 0.021092% for the three-battery instance. These results show that the proposed metaheuristic is able to provide solutions very close to the exact reference values.

The CPU time results also showed the computational interest of the proposed method. HRC-ALNS often reached its best solution before the end of the search, while the remaining iterations were used to confirm the absence of further improvement. For larger instances involving four and five battery types, where no exact MILP solution was available, the proposed method was still able to generate feasible and interpretable solutions. This confirms the added value of HRC-ALNS as a scalable solution approach for larger disassembly line instances.

Limitations and Perspectives

Despite the obtained results, some limitations remain. First, the computational experiments were conducted on a limited set of battery instances. Although these instances were sufficient to validate the proposed method and analyse its behaviour, additional tests on larger and more diverse industrial data would strengthen the generality of the conclusions.

Second, the stochastic version considers uncertainty only in human operation times. This is an important source of variability in human-robot collaborative disassembly, but it is not the only one. In real disassembly systems, uncertainty may also affect product condition, task feasibility, component accessibility, robot availability, tool reliability, and safety conditions. Future work could extend the stochastic model to include these additional sources of uncertainty.

Third, the proposed objective function is mono-objective and mainly cost-oriented. This choice makes the evaluation clear and operational, but other criteria could also

be relevant. For example, ergonomic risk, safety exposure, workload balance, energy consumption, environmental impact, or recovery value could be integrated into future formulations. A multi-objective extension would make it possible to analyse trade-offs between cost, safety, robustness, and operational performance.

Fourth, the proposed HRC-ALNS was evaluated using predefined parameters. Although these parameters provided good results in the tested instances, a more systematic calibration could improve the performance of the algorithm. Future work could include parameter tuning methods or adaptive parameter control during the search.

Several perspectives can therefore be proposed. A first direction is to test the proposed method on additional battery architectures and larger industrial-scale instances. In particular, future work will consider a higher number of battery types in order to evaluate the scalability of the proposed HRC-ALNS on more complex multi-product disassembly lines. A second direction is to improve the stochastic modelling by increasing the number of scenarios used to represent human operation time variability. Working with a larger scenario set would make it possible to better evaluate the robustness of the obtained solutions under uncertainty. A third direction is to extend the method toward a multi-objective framework in order to analyse trade-offs between cost, safety, workload balance and robustness. Another possible extension is to combine the proposed metaheuristic with simulation-based evaluation in order to analyse dynamic aspects of the disassembly line, such as waiting times, resource availability, and operational disturbances.

In future work, the objective will therefore be to extend the computational study to a larger number of batteries and to a richer set of stochastic scenarios, in order to further test the robustness, scalability and industrial relevance of the proposed HRC-ALNS methodology.

In conclusion, this work shows that metaheuristic optimization can provide an effective and flexible solution approach for complex human-robot collaborative battery disassembly line problems. Exact MILP models remain valuable for defining the problem rigorously and validating small instances, but the proposed HRC-ALNS offers a more scalable alternative for larger cases. By combining problem-specific encoding, decoding, adaptive neighbourhood search, and stochastic evaluation, the methodology developed in this thesis contributes to the optimization of future flexible and collaborative disassembly systems.

Bibliography

- [1] Zekai Ai, A. Y. C. Nee, and S. K. Ong. Artificial intelligence in electric vehicle battery disassembly: A systematic review. *Automation*, 5(4):28, 2024. <https://doi.org/10.3390/automation5040028>.
- [2] M. Alfaro-Algaba and F. Javier Ramirez. Techno-economic and environmental disassembly planning of lithium-ion electric vehicle battery packs for remanufacturing. *Resources, Conservation and Recycling*, 154:104461, 2020. <https://doi.org/10.1016/j.resconrec.2019.104461>.
- [3] Sabri Baazouzi, Felix Paul Rist, Max Weeber, and Kai Peter Birke. Optimization of disassembly strategies for electric vehicle batteries. *Batteries*, 7(4):74, 2021. <https://doi.org/10.3390/batteries7040074>.
- [4] Olga Battaïa and Alexandre Dolgui. A taxonomy of line balancing problems and their solution approaches. *International Journal of Production Economics*, 142(2):259–277, 2013. <https://doi.org/10.1016/j.ijpe.2012.10.020>.
- [5] Christian Becker and Armin Scholl. A survey on problems and methods in generalized assembly line balancing. *European Journal of Operational Research*, 168(3):694–715, 2006. <https://doi.org/10.1016/j.ejor.2004.07.023>.
- [6] Giovanni Belingardi and Alessandro Scattina. Battery pack and underbody: Integration in the structure design for battery electric vehicles—challenges and solutions. *Vehicles*, 5(2):498–514, 2023. <https://doi.org/10.3390/vehicles5020028>.
- [7] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. *Robust Optimization*. Princeton University Press, Princeton, 2009. <https://press.princeton.edu/books/hardcover/9780691143682/robust-optimization>.
- [8] Dimitris Bertsimas and John N. Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific, Belmont, MA, 1997. <http://athenasc.com/linoptbook.html>.
- [9] John R. Birge and François Louveaux. *Introduction to Stochastic Programming*. Springer, New York, 2 edition, 2011. <https://doi.org/10.1007/978-1-4614-0237-4>.
- [10] Nils Boysen, Malte Fliedner, and Armin Scholl. A classification of assembly line balancing problems. *European Journal of Operational Research*, 183(2):674–693, 2007. <https://doi.org/10.1016/j.ejor.2006.10.010>.
- [11] T. C. E. Cheng, B. M. T. Lin, and Y. Tian. A scheduling model for the refurbishing process in recycling management. *International Journal of Production Research*, 51(23–24):7120–7139, 2013. <https://doi.org/10.1080/00207543.2013.842024>.
- [12] Mengling Chu and Weida Chen. Human-robot collaboration disassembly planning for end-of-life power batteries. *Journal of Manufacturing Systems*, 69:271–291, 2023. <https://doi.org/10.1016/j.jmsy.2023.06.014>.
- [13] Marco Dorigo, Mauro Birattari, and Thomas Stützle. Ant colony optimization. *IEEE Computational Intelligence Magazine*, 1(4):28–39, 2006. <https://doi.org/10.1109/MCI.2006.329691>.

-
- [14] Franz Ehm. Scheduling and process planning for the dismantling shop with flexible disassembly mode and recovery level. *Computers & Industrial Engineering*, 190:109927, 2024. <https://doi.org/10.1016/j.cie.2024.109927>.
- [15] Yaping Fu, MengChu Zhou, Xiwang Guo, and Liang Qi. Stochastic multi-objective integrated disassembly-reprocessing-reassembly scheduling via fruit fly optimization algorithm. *Journal of Cleaner Production*, 278:123364, 2021. <https://doi.org/10.1016/j.jclepro.2020.123364>.
- [16] Jiaxu Gao, Guoxian Wang, Jinhua Xiao, Pai Zheng, and Eujin Pei. Partially observable deep reinforcement learning for multi-agent strategy optimization of human-robot collaborative disassembly: A case of retired electric vehicle battery. *Robotics and Computer-Integrated Manufacturing*, 89:102775, 2024. <https://doi.org/10.1016/j.rcim.2024.102775>.
- [17] Fred Glover. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research*, 13(5):533–549, 1986. [https://doi.org/10.1016/0305-0548\(86\)90048-1](https://doi.org/10.1016/0305-0548(86)90048-1).
- [18] Aşkiner Güngör and Surendra M. Gupta. A solution approach to the disassembly line balancing problem in the presence of task failures. *International Journal of Production Research*, 39(7):1427–1467, 2001. <https://doi.org/10.1080/00207540110052157>.
- [19] Lei Guo, Zeqiang Zhang, Haolin Song, and Yan Li. Industrial application of a human-robot collaborative parallel two-sided destructive disassembly line balancing problem in multi-product, multi-line layouts. *Robotics and Computer-Integrated Manufacturing*, page 103209, 2026. <https://doi.org/10.1016/j.rcim.2025.103209>.
- [20] Xiwang Guo, Zhiliang Bi, Jiacun Wang, Shujin Qin, Shixin Liu, and Liang Qi. Reinforcement learning for disassembly system optimization problems: A survey. *International Journal of Network Dynamics and Intelligence*, 2(1), 2023. <https://doi.org/10.53941/ijndi0201001>.
- [21] Xiwang Guo, Chenyang Fan, Mengchu Zhou, Shixin Liu, Jiacun Wang, Shujin Qin, and Ying Tang. Human-robot collaborative disassembly line balancing problem with stochastic operation time and a solution via multi-objective shuffled frog leaping algorithm. *IEEE Transactions on Automation Science and Engineering*, 21(3):4448–4459, 2024. <https://doi.org/10.1109/TASE.2023.3296733>.
- [22] Muyue Han, Lingxiang Yun, and Lin Li. Deep reinforcement learning-based approach for dynamic disassembly scheduling of end-of-life products with stimuli-activated self-disassembly. *Journal of Cleaner Production*, page 138758, 2023. <https://doi.org/10.1016/j.jclepro.2023.138758>.
- [23] Gavin Harper, Roberto Sommerville, Emma Kendrick, Laura Driscoll, Peter Slater, Rustam Stolkin, Allan Walton, Paul Christensen, Oliver Heidrich, Simon Lambert, Andrew Abbott, Karl Ryder, Linda Gaines, and Paul Anderson. Recycling lithium-ion batteries from electric vehicles. *Nature*, 575:75–86, 2019. <https://doi.org/10.1038/s41586-019-1682-5>.
- [24] John H. Holland. *Adaptation in Natural and Artificial Systems*. MIT Press, Cambridge, MA, 1992.

-
- [25] Tero Kaarlela, Nina Siltala, Kimmo Valtanen, Tapio Heikkilä, and Minna Lanz. Robotised disassembly of electric vehicle batteries: A systematic literature review. *Journal of Manufacturing Systems*, 74:901–921, 2024. <https://doi.org/10.1016/j.jmsy.2024.05.013>.
- [26] Achim Kampker, Heiner Hans Heimes, Christian Offermanns, Janis Vienenkötter, and Tobias Robben. Framework and classification of battery system architectures. *World Electric Vehicle Journal*, 14(4):88, 2023. <https://doi.org/10.3390/wevj14040088>.
- [27] Qingdi Ke, Peng Zhang, Lei Zhang, and Shouxu Song. Electric vehicle battery disassembly sequence planning based on frame-subgroup structure combined with genetic algorithm. *Frontiers in Mechanical Engineering*, 6:576642, 2020. <https://doi.org/10.3389/fmech.2020.576642>.
- [28] Scott Kirkpatrick, C. Daniel Gelatt, and Mario P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983. <https://doi.org/10.1126/science.220.4598.671>.
- [29] Xin Li, Mengchu Zhou, Shixin Liu, Jiacun Wang, and Xiwang Guo. Artificial intelligence-driven end-of-life electric vehicle battery disassembly: A review. *Robotics and Computer-Integrated Manufacturing*, 88:102744, 2024. <https://doi.org/10.1016/j.rcim.2024.102744>.
- [30] Pei Liang, Yaping Fu, and Kaizhou Gao. Multi-product disassembly line balancing optimization method for high disassembly profit and low energy consumption with noise pollution constraints. *Engineering Applications of Artificial Intelligence*, page 107721, 2024. <https://doi.org/10.1016/j.engappai.2023.107721>.
- [31] Jia Mao, Shenggang Li, Ziang Zhao, Yanzhi Zhou, Jinyuan Cheng, and Weiwen Li. Optimization of dismantling line for used electric vehicle battery packs. *Symmetry*, 17(6):916, 2025. <https://doi.org/10.3390/sym17060916>.
- [32] Ronghua Meng, Yuxiang Tian, Ningjing Shu, Liuyang Cai, Huawei Song, and Yuping Zhang. Disassembly sequence optimization of lithium battery pack based on priority constraint graph. *Journal of Physics: Conference Series*, 2650(1):012021, 2023. <https://doi.org/10.1088/1742-6596/2650/1/012021>.
- [33] Kaisa Miettinen. *Nonlinear Multiobjective Optimization*. Springer, Boston, MA, 1999. <https://doi.org/10.1007/978-1-4615-5563-6>.
- [34] Elena Mossali, Nicoletta Picone, Luca Gentilini, Olga Rodriguez, Juan Manuel Perez, and Marcello Colledani. Lithium-ion batteries towards circular economy: A literature review of opportunities and issues of recycling treatments. *Journal of Environmental Management*, 264:110500, 2020. <https://doi.org/10.1016/j.jenvman.2020.110500>.
- [35] Alessandro Neri, Maria Angela Butturi, and Rita Gamberini. Sustainable management of electric vehicle battery remanufacturing: A systematic literature review and future directions. *Journal of Manufacturing Systems*, 2024. <https://doi.org/10.1016/j.jmsy.2024.10.006>.

-
- [36] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, New York, 2 edition, 2006. <https://doi.org/10.1007/978-0-387-40065-5>.
- [37] Eren Özceylan, Can B. Kalayci, Aşkiner Güngör, and Surendra M. Gupta. Disassembly line balancing problem: A review of the state of the art and future directions. *International Journal of Production Research*, 57(15–16):4805–4827, 2019. <https://doi.org/10.1080/00207543.2018.1428775>.
- [38] Michael L. Pinedo. *Scheduling: Theory, Algorithms, and Systems*. Springer, Cham, 5 edition, 2016. <https://doi.org/10.1007/978-3-319-26580-3>.
- [39] Shujin Qin, Changsheng Xiang, Jiacun Wang, Shixin Liu, Xiwang Guo, and Liang Qi. Multiple product hybrid disassembly line balancing problem with human–robot collaboration. *IEEE Transactions on Automation Science and Engineering*, 2024. <https://doi.org/10.1109/TASE.2024.3418898>.
- [40] Max Rettenmeier, Mauritz Moller, and Alexander Sauer. Disassembly technologies of end-of-life automotive battery packs as the cornerstone for a circular battery value chain: A process-oriented analysis. *Resources, Conservation and Recycling*, 209:107786, 2024. <https://doi.org/10.1016/j.resconrec.2024.107786>.
- [41] Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczyński. *Lectures on Stochastic Programming: Modeling and Theory*. SIAM, Philadelphia, 2009. <https://doi.org/10.1137/1.9780898718751>.
- [42] Hamdy A. Taha. *Operations Research: An Introduction*. Pearson, Harlow, 10 edition, 2019. <https://www.pearson.com/en-gb/subject-catalog/p/operations-research-an-introduction-global-edition/P200000005533/9781292165561>.
- [43] Guangdong Tian, Wenjie Wang, Honghao Zhang, Xiaowan Zhou, Cheng Zhang, and Zhiwu Li. Multi-objective optimization of energy-efficient remanufacturing system scheduling problem with lot-streaming production mode. *Expert Systems with Applications*, 237:121309, 2024. <https://doi.org/10.1016/j.eswa.2023.121309>.
- [44] Enrico Villagrossi and Tito Dinon. Robotics for electric vehicles battery packs disassembly towards sustainable remanufacturing. *Journal of Remanufacturing*, 13:355–379, 2023. <https://doi.org/10.1007/s13243-023-00134-z>.
- [45] Laurence A. Wolsey. *Integer Programming*. John Wiley & Sons, New York, 1998.
- [46] Shubiao Wu, Nicolaj Kaden, and Klaus Dröder. A systematic review on lithium-ion battery disassembly processes for efficient recycling. *Batteries*, 9(6):297, 2023. <https://doi.org/10.3390/batteries9060297>.
- [47] Tengfei Wu, Zeqiang Zhang, Lei Guo, Haoxuan Song, Xinlan Xie, and Shiyi Ren. A hybrid evolutionary algorithm for the stochastic human–robot collaborative disassembly line balancing problem considering carbon emission optimization. *Engineering Applications of Artificial Intelligence*, page 108703, 2024. <https://doi.org/10.1016/j.engappai.2024.108703>.
- [48] Tengfei Wu, Zeqiang Zhang, Tao Yin, and Yu Zhang. Multi-objective optimisation for cell-level disassembly of waste power battery modules in human-machine hybrid

-
- mode. *Waste Management*, 144:513–526, 2022. <https://doi.org/10.1016/j.wasman.2022.04.015>.
- [49] Tengfei Wu, Zeqiang Zhang, Yanqing Zeng, Yu Zhang, Lei Guo, and Junqi Liu. Techno-economic and environmental benefits-oriented human–robot collaborative disassembly line balancing optimization in remanufacturing. *Robotics and Computer-Integrated Manufacturing*, 86:102650, 2024. <https://doi.org/10.1016/j.rcim.2023.102650>.
- [50] Shibo Yang, Xiaojun Zhuo, Wei Ning, Xing Xia, and Yong Huang. Integrated risk-aware smart disassembly planning for scrap electric vehicle batteries. *Energies*, 17(12):2946, 2024. <https://doi.org/10.3390/en17122946>.
- [51] Lifeng Yin, Chuang Wang, Liang Cong, and Quanyi Du. A sequential disassembly planning approach based on knowledge graph and graph isomorphism network for supporting power battery remanufacturing. *Journal of Cleaner Production*, 507:145558, 2025. <https://doi.org/10.1016/j.jclepro.2025.145558>.
- [52] Gang Yuan, Yinsheng Yang, Guangdong Tian, and Amir Mohammad Fathollahi-Fard. Capacitated multi-objective disassembly scheduling with fuzzy processing time via a fruit fly optimization algorithm. Research Square preprint, 2021. <https://doi.org/10.21203/rs.3.rs-1075510/v1>.
- [53] Changshu Zhan, Xuesong Zhang, Guangdong Tian, Duc Truong Pham, Mikhail Ivanov, Anatoly Aleksandrov, Chenxi Fu, Junnan Zhang, and Zhen Wu. Environment-oriented disassembly planning for end-of-life vehicle batteries based on an improved northern goshawk optimisation algorithm. *Environmental Science and Pollution Research*, 2023. <https://doi.org/10.1007/s11356-023-25599-0>.
- [54] Wenkang Zhang, Yufan Zheng, and Rafiq Ahmad. The integrated process planning and scheduling of flexible job-shop-type remanufacturing systems using improved artificial bee colony algorithm. *Journal of Intelligent Manufacturing*, 34(7):2963–2988, 2022. <https://doi.org/10.1007/s10845-022-01969-2>.
- [55] Wenkang Zhang, Yufan Zheng, and Rafiq Ahmad. An energy-efficient multi-objective integrated process planning and scheduling for a flexible job-shop-type remanufacturing system. *Advanced Engineering Informatics*, 56:102010, 2023. <https://doi.org/10.1016/j.aei.2023.102010>.
- [56] Wenkang Zhang, Yufan Zheng, and Rafiq Ahmad. An energy-efficient multi-objective scheduling for flexible job-shop-type remanufacturing system. *Journal of Manufacturing Systems*, 66:211–232, 2023. <https://doi.org/10.1016/j.jmsy.2022.12.008>.
- [57] Xuesong Zhang, Hao Zhou, Chenxi Fu, Menghan Mi, Changshu Zhan, Duc Truong Pham, and Amir M. Fathollahi-Fard. Application and planning of an energy-oriented stochastic disassembly line balancing problem. *Environmental Science and Pollution Research*, 2023. <https://doi.org/10.1007/s11356-023-27288-4>.
- [58] Xugang Zhang, Hengheng Xu, Qingshan Gong, and Yan Wang. Multi-objective human-robot collaboration disassembly planning for retired power batteries based on entropy-weight fuzzy hierarchical analysis and differential evolution algorithm. *Expert Systems with Applications*, page 131697, 2026. <https://doi.org/10.1016/j.eswa.2026.131697>.

-
- [59] Zeqiang Zhang, Yue Han, Yan Li, Yanqing Zeng, Qiqi Xia, and Kaipu Wang. Reliability and efficiency optimization of multi-product disassembly lines with destructive operations and preventive maintenance. *Reliability Engineering & System Safety*, 272:112525, 2026. <https://doi.org/10.1016/j.ress.2026.112525>.
- [60] Dong Zhu, Xuesong Zhang, Xinyue Huang, Duc Truong Pham, and Changshu Zhan. A stochastic sequence-dependent disassembly line balancing problem with an adaptive large neighbourhood search algorithm. *Processes*, 13(6):1675, 2025. <https://doi.org/10.3390/pr13061675>.
- [61] Lixia Zhu, Yarong Chen, and Jabir Mumtaz. Multi-objective human-robot collaborative disassembly line balancing considering components remanufacture demand and hazard characteristics. *Computers & Industrial Engineering*, page 110621, 2024. <https://doi.org/10.1016/j.cie.2024.110621>.