University of Nottingham
UK | CHINA | MALAYSIA

Asset Selection and Optimisation for Robotic Assembly Cell Reconfiguration

Thesis submitted to the University of Nottingham for the degree of
Doctor of Philosophy, November 2023.

Fan Mo

Student ID: 20303223

Signature:

Date: 01/11/2023

Abstract

With the development of Industry 4.0, the manufacturing industry has revolutionised a lot. Product manufacture become more and more customised. This trend is achieved by innovative techniques, such as the reconfigurable manufacturing system (RMS). This system is designed at the outset for rapid change in its structure, as well as in software and hardware components, to respond to market changes quickly. Robots are important in these systems because they provide the agility and precision required to adapt rapidly to new manufacturing processes and customisation demands.

Despite the importance of applying robots in these systems, there might be some challenges. For example, there is data from multiple sources, such as the technical manual sensor data. Besides, robot applications must react quickly to the ever-changing process requirements to meet customer's requirements. Furthermore, further optimisation, especially layout optimisation, is needed to ensure production efficiency after adaptation to the current process requirements.

To address these challenges, this doctoral thesis presents a framework for reconfiguring robotic assembly cells in manufacturing. This framework consists of three parts: the experience databank, the methodology for optimal manufacturing asset selection, and the methodology for layout optimisation.

The experience databank is introduced to confront the challenge of assimilating and processing heterogeneous data from numerous manufacturing sources, which is achieved by proposing a vendor-neutral ontology model. This model is specifically designed for encapsulating information about robotic assembly cells and is subsequently applied to a knowledge graph. The resulting knowledge graph, constituting the experience databank, facilitates the effective organisation and interpretation of the diverse data.

An optimal manufacturing asset selection methodology is introduced to adapt to shifting processes and product requirements, which focuses on identifying potential assets and their subsequent evaluation. This approach integrates a modular evaluation framework that considers multiple criteria such as cost, energy consumption, and robot manoeuvrability, ensuring the selection process remains robust in changing market demands and product requirements.

A scalable methodology for layout optimisation within the reconfigurable robotic assembly cells is proposed to resolve the need for further optimisation post-adaption. It introduces a scalable, multi-decision modular optimisation framework that synergises a simulation environment, optimisation environment, and robust optimisation algorithms. This strategy utilises the insights garnered from the experience databank to facilitate informed decision-making, thereby enabling the robotic assembly cells to not only meet the immediate production exigencies but also align with the manufacturing landscape's evolving dynamics.

The validation of the three methodologies presented in this doctoral thesis encompasses both software development and practical application through three distinct use cases. For the experience databank, an interface was developed using Protégé, Neo4j, and Py2neo, allowing for effective organisation and processing of varied manufacturing data. The programming interface for the asset selection methodology was built using Python, integrating with the experience databank via Py2neo and Neo4j to facilitate dynamic and informed decision-making in asset selection. In terms of software for the layout optimisation framework, two different applications were developed to demonstrate the framework's scalability and adaptability. The first application, combining Python and C# programming with Siemens Tecnomatix Process Simulate, is geared towards optimising layouts involving multiple machines. The second application utilises Python programming alongside the RoboDK API and RoboDK software, tailored for layout optimisation in scenarios involving a single robot.

Complementing these software developments, the methodologies were further validated through three use cases, each addressing a unique aspect of the framework. Use Case 1 focused on implementing asset selection and system layout optimisation based on a single objective, leveraging the experience databank. The required assets are selected, and the required cycle time for executing the whole robotic assembly operation has been reduced by 15.6% from 47.17 seconds to 39.83 seconds. Use Case 2 extended the layout optimisation to single-robot operations with an emphasis on multi-criteria decision-making. The energy consumption was minimised to

5613.59 Wh after implementing optimisation strategies, demonstrating a significant enhancement in energy efficiency. Compared with the baseline of 6164.98 Wh, this represents an 8.9% reduction in energy usage. For minimised cycle time, a reduction of 6.0% from the baseline of 57.11 s is achieved, resulting in a cycle time of 53.15 s. Regarding the pursuit of a maximised robot manoeuvrability index, an increase of 140.8% from the baseline of 0.4891235 is achieved, resulting in a maximised value of 1.1786125. Lastly, Use Case 3 tested the modular and multi-objective asset selection methodology, demonstrating its efficacy across diverse operational scenarios. Evaluations conducted with two multi-objective optimisation algorithms, Non-Dominated Sorting Genetic Algorithm II (NSGA-II) and Strength Pareto Evolutionary Algorithm II (SPEA-II), revealed interesting implications for selecting and optimising robotic assets in response to new customer requests. Specifically, SPEA-II identified a Pareto solution that was more cost-effective (£20,920) compared to NSGA-II (£21,090), while maintaining a competitive specification efficiency score (0.865 vs. 0.879). Consequently, SPEA-II is preferred for optimising robotic asset selection in scenarios prioritising cost. However, should the requirement shift towards maximising specification efficiency, the NSGA-II would be the more suitable choice.

These use cases not only showcased the practical applicability of the developed software but also underlined the robustness and adaptability of the proposed methodologies in real-world manufacturing environments.

In conclusion, this doctoral thesis presents a methodology for reconfiguring robotic assembly cells in manufacturing. By harnessing the capabilities of artificial intelligence, knowledge graphs, and simulation methodologies, it addresses the challenges of processing data from diverse sources, adapting to fluctuating market demands, and establishing further optimisations for enhanced operational efficiency in the modern manufacturing landscape. To affirm the viability of this framework, the thesis integrates software development procedures tailored to the proposed methodologies and furnishes evidence through three use cases, which are evaluated against well-defined criteria.

List of Publications

This section outlines the research papers generated to contribute to the PhD research.

- Paper 1: Mo, Fan; Chaplin, Jack C.; Sanderson, David; Martínez-Arellano, Giovanna; Ratchev, Svetan. Semantic Models and Knowledge Graphs as Manufacturing System Reconfiguration Enablers. *Robotics and Computer-Integrated Manufacturing*, 86:102625, April 2024. https://doi.org/10.1016/j.rcim.2023.102625
- Paper 2: Mo, Fan; Chaplin, Jack C.; Sanderson, David; Rehman, Hamood Ur; Monetti, Fabio Marco; Maffei, Antonio; Ratchev, Svetan. A Framework for Manufacturing System Reconfiguration Based on Artificial Intelligence and Digital Twin. In *FAIM 2022: Flexible Automation and Intelligent Manufacturing: The Human-Data-Technology Nexus*, pp 361–373. https://doi.org/10.1007/978-3-031-18326-3_35
- Paper 3: Martínez-Arellano, Giovanna; Niewiadomski, Karol; Mo, Fan; Elshafei, Basem; Chaplin, Jack C.; Mcfarlane, Duncan; Ratchev, Svetan. Enabling Coordinated Elastic Responses of Manufacturing Systems through Semantic Modelling. *IFAC World Congress 2023*, 56(2):7402-7407, 2023. https://doi.org/10.1016/j.ifacol.2023.10.617

Paper 1 contributed to the development of Chapter 4 and Chapter 8, describing the methodology for building the experience databank via the ontology model and knowledge graph.

Paper 2 contributed to the development of Chapter 3, which describes the overview of my proposed methodology, and Chapter 6, which describes the layout optimisation framework.

Paper 3 is a joint paper between the University of Nottingham and the University of Cambridge. The semantic content in this paper contributed to the development of Chapter 4 of the PhD thesis.

Acknowledgement

My sincerest appreciation and profound admiration are directed towards my distinguished and esteemed supervisors: Professor Svetan Ratchev, Doctor Jack C. Chaplin, Doctor David Sanderson, and Professor Atanas Popov. Throughout the exhilarating journey of my doctoral research, their unwavering guidance, exceptional expertise, and unparalleled mentorship have proven invaluable. Their wisdom has shone a light on my path, and their tireless support has genuinely facilitated both my academic and personal growth. Their insightful criticisms and timely encouragement have moulded my doctoral research and significantly influenced my worldview and intellectual perspective.

Their dedication to their research field and their unyielding quest for excellence have consistently served as an immeasurable source of inspiration. The depth of their knowledge, the breadth of their experience, and their generosity in sharing these have been pivotal to my scholarly development. I am deeply indebted to them for the numerous opportunities for collaboration on invigorating projects and for championing the publication of impactful research papers.

It is with profound privilege that I express my deepest appreciation to my beloved wife, Xue Yang, who has been a constant pillar of strength and a beacon of support throughout this intense intellectual voyage. Her boundless patience, ongoing encouragement, and unwavering love have underpinned my achievements. I remain eternally grateful for her steadfast dedication and her endless belief in my potential.

To my treasured parents, Pai Mo and Ming Deng, words fall short of conveying the depth of my gratitude. Their unconditional love, relentless support, and tireless dedication to my well-being have been my guiding light. Their unwavering faith in me, even in the face of immense challenges, has bolstered my resilience and determination. My aspiration is to continuously

make them proud as I tread the path they've illuminated for me.

I am deeply grateful to my esteemed colleagues at the University of Nottingham — Giovanna, Alison, Kryssa, Nancy, Helena, Karina, Kerri, Basem, Karol, Chengyuan, Sara, Likun, Pete, Kev, Joe H., Joe G., James and Mark — for their invaluable camaraderie. Their insightful feedback and stimulating discussions have significantly enriched my academic and personal growth. The countless shared moments of grappling with complex ideas, spirited debates over theories, celebrations of our triumphs, and collective efforts to overcome challenges have been pivotal in shaping my doctoral journey. The bonds of friendship forged amidst these intellectually vibrant exchanges are treasured and will indubitably endure, holding a cherished place in my heart.

I must convey my heartfelt appreciation to the DiManD Innovative Training Network (ITN) project, which has supported my research through funding from the European Union via the Marie Skłodowska-Curie Innovative Training Networks (H2020-MSCA-ITN-2018) under grant agreement No. 814078. Besides, I must also express my thanks to all the DiManD ESRs: Agajan, Hamood, Miriam, Fabio, Hien, Angela, Jose, Trunal, Luis, Terrin, Nathaly, and Joaquín. My gratitude extends to the coordinator of the DiManD project, Miren, and the supervisors during my secondment in the DiManD project: Professor Barata, Antonio, Claudia, Lara, and Professor Fassi. Their dedication to cultivating an atmosphere that promotes intellectual growth, kindles curiosity and spurs innovation is commendable. Being a part of this dynamic academic community has enriched my doctoral journey, and the memories forged will be cherished.

To every individual who has been a part of this journey, I extend my deepest thanks. Your unwavering support, continuous encouragement, and shared insights have been the driving force behind my endeavours. As I embark on new academic and personal ventures, I carry with me the invaluable lessons imbibed, the cherished memories formed, and the enduring love and support received during this pivotal phase of my life.

Contents

Abstract i

List of Publications iv

Acknowledgements v

Table of Contents vii

List of Figures xi

List of Tables xiv

Nomenclature xvi

Abbreviations . . . xvi

Symbols . . . xix

Chapter 1 Introduction 1

1.1 Background and Motivation . . . 1

1.2 Aim and Objectives . . . 3

1.2.1 Research Questions . . . 3

1.2.2 Aim . . . 4

1.2.3 Objectives . . . 4

1.3 Thesis Structure . . . 5

Chapter 2 Literature Review 8

2.1 Reconfigurable Manufacturing Systems and Robotic Assembly . . . 10

2.1.1 Reconfigurable Manufacturing Systems . . . 10

2.1.2 Robotic Assembly in Reconfigurable Manufacturing Systems . . . 10

2.2 Ontology Models in the Manufacturing Domain . . . 11

2.3 Knowledge Graph . . . 13

2.3.1 Applications in Manufacturing . . . 14

2.3.2 Tools to Implement a Knowledge Graph . . . 15

2.4 Asset Selection in Robotic Assembly Cells . . . 17

2.5 Layout Optimisation in Robotic Assembly Cells . . . 19

2.6 Simulation Software for Robotic Assembly Cells . . . 20
2.7 Denavit-Hartenberg Parameters . . . 22
2.7.1 Standard Denavit-Hartenberg Parameters . . . 23
2.7.2 Modified Denavit-Hartenberg Parameters . . . 25
2.8 Multi-Objective Optimisation . . . 26
2.9 Chapter Summary . . . 29
Chapter 3 Reconfiguration Framework 31
3.1 The Proposed Methodology . . . 32
3.2 Validation Methods . . . 36
3.2.1 Validation of Methodology for Building and Updating Experience Databank . . . 36
3.2.2 Validation of Optimal Manufacturing Asset Selection Methodology . . . 37
3.2.3 Validation of Methodology for Layout Optimisation . . . 38
3.3 Chapter Summary . . . 39
Chapter 4 Ontology Model and Experience Databank 41
4.1 Introduction . . . 41
4.2 Development of Ontology Model and Experience Databank in Knowledge Graph 42
4.2.1 Knowledge Graph Building Approach . . . 43
4.2.2 Ontology Model as the Schema Layer of the Knowledge Graph . . . 43
4.2.3 Construction of the Entity Layer of the Knowledge Graph . . . 64
4.2.4 Application of the Generated Knowledge Graph (Experience Databank) . 66
4.3 Chapter Summary . . . 67
Chapter 5 Optimal Manufacturing Asset Selection 69
5.1 Introduction . . . 69
5.2 Methodology for Optimal Manufacturing Asset Selection . . . 70
5.3 Methodology for Identifying Candidate Assets . . . 71
5.3.1 Identifying Candidate Assets Based on Static Information . . . 73
5.3.2 Enhancing the Result Based on the Capacity Information . . . 75
5.4 Methodology for Evaluating Candidate Assets . . . 76
5.4.1 Introduction . . . 76
5.4.2 Evaluators Definition . . . 77
5.4.3 Evaluator Weights in Asset Evaluation . . . 87
5.5 Chapter Summary . . . 91

Chapter 6 Layout Configuration Optimisation 92
6.1 Introduction . . . 92
6.2 Optimisation Framework for Robotic Assembly Cell . . . 93
6.2.1 Simulation Environment . . . 95
6.2.2 Optimisation Environment . . . 95
6.2.3 Optimisation Algorithms . . . 96
6.3 Chapter Summary . . . 109
Chapter 7 Software Implementation 110
7.1 Introduction . . . 110
7.2 Interface Development for Experience Databank . . . 111
7.2.1 Software selection . . . 111
7.2.2 Development Process . . . 112
7.3 Interface Development for Asset Selection . . . 115
7.3.1 Software Selection for Asset Selection . . . 115
7.3.2 Development Process . . . 117
7.4 Implementing Layout Optimisation . . . 118
7.4.1 Software Selection for Layout Optimisation . . . 119
7.4.2 Simulation Environment Interface Development . . . 119
7.4.3 Optimisation Environment . . . 122
7.4.4 Optimisation Algorithm . . . 123
7.5 Chapter Summary . . . 124
Chapter 8 Use Case Studies 125
8.1 Use Case 1 – Asset Selection and System Layout Optimisation Using the Experience Databank . . . 125
8.1.1 Introduction . . . 126
8.1.2 Construction of the Knowledge Graph . . . 127
8.1.3 Application of the Experience Databank . . . 128
8.1.4 Discussion of the Results of Use Case 1 . . . 144
8.2 Use Case 2 – Multi-Criteria Layout Optimisation in a Single Robot via Experience Databank . . . 147
8.2.1 Introduction . . . 148
8.2.2 Establishment of the Experience Databank . . . 148
8.2.3 Layout Configuration Optimisation . . . 150

8.2.4 Discussion of the Results of Use Case 2 . . . 166
8.3 Use Case 3 – Modular Multi-Objective Asset Selection with Experience Databank 168
8.3.1 Introduction . . . 168
8.3.2 Generation of the Experience Databank . . . 169
8.3.3 Modular Asset Selection Utilising Experience Databank . . . 170
8.3.4 Discussion of the Results of Use Case 3 . . . 185
8.4 Chapter Summary . . . 187
Chapter 9 Conclusions and Future Work 189
9.1 Conclusions . . . 189
9.2 Applications in the Manufacturing Industry . . . 193
9.2.1 Automotive Industry . . . 193
9.2.2 Aerospace Manufacturing . . . 194
9.3 Future Work . . . 195
References 197

List of Figures

Figure 1.1: Structure of this PhD thesis 6

Figure 2.1: Structure of the literature review chapter 8
Figure 2.2: Standard DH parameters 24
Figure 2.3: Modified DH parameters 25

Figure 3.1: Schematic representation of the research's key components 31
Figure 3.2: The three contributions to robotic assembly cell reconfiguration 32
Figure 3.3: Unified Modeling Language (UML) activity diagram of the reconfiguration framework 33

Figure 4.1: Relationship between experience databank, knowledge graph and ontology model 42
Figure 4.2: Linking semantic models in the optimisation and the decision-making process for the reconfiguration 45
Figure 4.3: Structure of the reconfiguration semantic model 49
Figure 4.4: Example to show the relationship between the two types of the process in the reconfiguration-related task 50
Figure 4.5: Robot semantic model 61
Figure 4.6: Required parameters for determining the end effector pose 62
Figure 4.7: Orientation definition based on different brands 63
Figure 4.8: FANUC J2 and J3 interaction 64
Figure 4.9: Process of construction of the entity layer of the knowledge graph . . . 65
Figure 4.10: Knowledge inference process in the experience databank (achieved by knowledge graph): a. Generation of the query. b. Ontology reasoning. c. Generation of the inference results based on the ontology reasoning . 67

Figure 5.1: Simplified manufacturing asset selection process. 71
Figure 5.2: Workflow of identifying the candidate assets 73

Figure 5.3: Simplified process for calculating the specification efficiency score 79
Figure 5.4: Detailed process of calculating the specification efficiency score 80
Figure 5.5: Procedure to determine the reconfiguration costs 86
Figure 5.6: Asset selection in the scenario of unknown weights of major evaluators . 90

Figure 6.1: Details of utilising the reconfiguration optimisation framework 94
Figure 6.2: Workspace of the FANUC ER-4iA robot 105
Figure 6.3: Updated workspace of the FANUC ER-4iA robot 108

Figure 7.1: UML activity diagram for building the experience databank 111
Figure 7.2: Ontology model in Protégé 113
Figure 7.3: Graph initialisation in Neo4j 114
Figure 7.4: Exemplification of a generated knowledge graph 115
Figure 7.5: UML activity diagram for optimal asset selection 116
Figure 7.6: UML activity diagram for layout configuration optimisation 118
Figure 7.7: Flowchart of using algorithms suggested from experience databank . . . 123

Figure 8.1: Validation components of the proposed methodology in Use Case 1 . . . 126
Figure 8.2: Physical layout of one of the OMNIFACTORY's test plants 126
Figure 8.3: Construction of the knowledge graph (experience databank) 128
Figure 8.4: Query command to find the required process for "Task 100" in Neo4j . . 129
Figure 8.5: Information about task 100 and its related nodes in the knowledge graph (experience databank) 129
Figure 8.6: Capability matching for "MarkingCapability" with the help of ontology reasoning 131
Figure 8.7: The capacity model enhances the resource selection process 132
Figure 8.8: Capability decomposition process for PickAndPlace capabiltiy 133
Figure 8.9: Finding candidate assets for "ForceApplying", "Moving", "Grasping" and "Releasing" capability 134
Figure 8.10: Detailed process about finding candidate assets for "ForceApplying", "Moving", "Grasping" and "Releasing" capability 136
Figure 8.11: Information about Task 50 137
Figure 8.12: Initial physical layout of the production cell from the Task 50 138
Figure 8.13: Reconfiguration model to enhance the resource selection process 139

Figure 8.14: Reconfiguration model to enhance the layout reconfiguration process . . 141
Figure 8.15: Recommendations from the knowledge graph to solve the layout optimisation problem . . . 142
Figure 8.16: Utilisation of the layout optimisation framework in the Use Case 1 . . . 143
Figure 8.17: Initial work cell layout . . . 144
Figure 8.18: Optimised work cell layout . . . 145
Figure 8.19: Optimisation fitness cure of the genetic algorithm . . . 145
Figure 8.20: Validation components of the proposed methodology . . . 147
Figure 8.21: Possible poses of doing the drilling operation . . . 149
Figure 8.22: Workpiece information . . . 150
Figure 8.23: Workflow of the layout optimisation in Use Case 2 . . . 150
Figure 8.24: Interface for the RoboDK . . . 151
Figure 8.25: Relationship between robot and workspace . . . 154
Figure 8.26: Orientation of the end effector pose . . . 155
Figure 8.27: Flowchart of recording energy consumption . . . 156
Figure 8.28: Flowchart of optimisation . . . 159
Figure 8.29: Baseline pose . . . 160
Figure 8.30: Pareto solutions obtained via NSGA-II . . . 161
Figure 8.31: Pareto solutions obtained via SPEA-II . . . 162
Figure 8.32: Pareto solutions obtained via NSGA-III . . . 163
Figure 8.33: Pareto solution (x_c: 302.96, y_c: 76.83, z_c: 303.63) with minimised energy consumption . . . 165
Figure 8.34: Pareto solution (x_c: 300.34, y_c: 87.62, z_c: 303.63) with minimised cycle time . . . 165
Figure 8.35: Pareto solution (x_c: 324.47, y_c: -0.12, z_c: 237.42) with maximised manoeuvrability index . . . 166
Figure 8.36: Focus of Use Case 3 . . . 168
Figure 8.37: Feature requirement for the new product . . . 172
Figure 8.38: Candidate assets information of the new Operation 1 . . . 173
Figure 8.39: Candidate assets information of the new Operation 2 . . . 175
Figure 8.40: Candidate assets information of the new Operation 3 . . . 175
Figure 8.41: Candidate assets information of the new Operation 4 . . . 176
Figure 8.42: Candidate assets information of the new Operation 5 . . . 176

Figure 8.43: Candidate assets information of the new Operation 6 . . . 177
Figure 8.44: Process for acquiring cost-related information . . . 180
Figure 8.45: Optimisation result of NSGA-II . . . 181
Figure 8.46: Pareto solutions of NSGA-II . . . 182
Figure 8.47: Optimisation result of SPEA-II . . . 183
Figure 8.48: Pareto solutions of SPEA-II . . . 184

List of Tables

Table 2.1: Comparison of Knowledge Graph Implementation Tools . . . 17
Table 4.1: Seven semantic models and their symbolic representation . . . 44
Table 5.1: Examples of criteria for selecting assets . . . 74
Table 5.2: Membership degrees as defined by the engineer . . . 84
Table 5.3: Changed optimisation objectives and targets . . . 90
Table 8.1: Combination asset information . . . 133
Table 8.2: Requirements . . . 138
Table 8.3: Candidate asset information . . . 139
Table 8.4: Specification efficiency score of each candidate asset under single evaluators . . . 140
Table 8.5: Total specification efficiency scores of each candidate asset . . . 140
Table 8.6: Validation according to the criteria in Use Case 1 . . . 146
Table 8.7: Modified Denavit-Hartenberg parameters of FANUC ER-4iA . . . 153
Table 8.8: Modified Denavit-Hartenberg parameters of FANUC ER-4iA with considering the length of the end effector . . . 153
Table 8.9: Pareto solutions found by NSGA-II . . . 161
Table 8.10: Pareto front found by SPEA-II . . . 162
Table 8.11: Pareto front found by NSGA-III . . . 164
Table 8.12: Validation according to the criteria in Use Case 2 . . . 166
Table 8.13: Capability information of “Product I” . . . 170

Table 8.14: Process requirements for producing "Product I" . . . 171
Table 8.15: Current assets to execute the operations . . . 171
Table 8.16: Related specification information of the current assets . . . 172
Table 8.17: Capability information of "Product II" . . . 173
Table 8.18: Process requirements for producing "Product II" . . . 174
Table 8.19: Optimisation objectives and targets . . . 179
Table 8.20: Specification evaluators for each operation . . . 179
Table 8.21: Comparison of hyperparameters in NSGA-II and SPEA-II . . . 181
Table 8.22: Reconfiguration cost details of NSGA-II . . . 182
Table 8.23: Specification efficiency score for the solution found by NSGA-II . . . 183
Table 8.24: Reconfiguration cost details of SPEA-II . . . 184
Table 8.25: Specification efficiency score for the solution found by SPEA-II . . . 185
Table 8.26: Validation according to the criteria in Use Case 3 . . . 186
Table 8.27: Summary of validated criteria in different use cases . . . 187

Nomenclature

Abbreviations

AC Asset Constraints.

AGV automated guided vehicles.

AHP Analytic Hierarchy Process.

AI artificial intelligence.

API Application Programming Interface.

APIs Application Programming Interfaces.

AR augmented reality.

ARC Area Constraint.

CAD Computer-Aided Design.

CEE Cyclic Event Evaluator.

CT Cycle Time.

DC Demand Constraint.

DH Denavit-Hartenberg.

EDD Earliest Due Date.

FIFO First In, First Out.

GC Grouping Constraint.

GPU graphics processing unit.

GUI graphical user interface.

HTTP Hypertext Transfer Protocol.

IC Investment Constraint.

IoT Internet of Things.

JC Job Constraints.

KPIs Key Performance Indicators.

LOD linked open data.

MC Machine Constraints.

MCDM multi-criteria decision-making.

MQTT Message Queuing Telemetry Transport.

MWR Most Work Remaining.

NC Non-Collision Constraint.

NRT Non-Reconfiguration-Related Task Model.

NSGA-II Non-Dominated Sorting Genetic Algorithm II.

NSGA-III Non-Dominated Sorting Genetic Algorithm III.

OCCR Ontology model of capability, capacity, and reconfiguration.

ONS Office for National Statistics.

OPC UA OPC Unified Architecture.

OWL Web Ontology Language.

PC Precedence Constraints.

PPE personal protective equipment.

PSL Process Specification Language.

RAC Resource Allocation Constraint.

RAMI Reference Architectural Model Industrie.

RC Reachability Constraint.

RDBMS relational database management system.

RDF Resource Description Framework.

RDFS RDF Schema.

RMS reconfigurable manufacturing system.

RMSs reconfigurable manufacturing systems.

RQ1 Research Question 1.

RQ2 Research Question 2.

RQ3 Research Question 3.

RT Reconfiguration-Related Task Model.

SC Space Constraint.

SPEA-II Strength Pareto Evolutionary Algorithm II.

SPT Shortest Processing Time.

SRR Specific Resource Requirements.

SU Space Utilisation.

TD Total Distance between Machines.

TOPSIS Technique for Order of Preference by Similarity to Ideal Solution.

TRA Total Resource Allocation.

UML Unified Modeling Language.

VR virtual reality.

XML Extensible Markup Language.

Symbols

Symbol	Description
A	Weight set of each sub-evaluator in the fuzzy evaluation
$a_1, a_2, \ldots, a_n$	Index list representing an index for each candidate group
B_i	Coefficient matrix representing the preferred relationship of assets under sub-evaluator U_i
B_1	Base or initial matrix for a specific decision or relationship in fuzzy logic or decision-making
b_{ef}^i	Coefficient of the preferred relationship of asset D_{me} to asset D_{mf} under sub-evaluator U_i
C	Candidate asset list (newly suggested assets)
$Cost_{Installation}$	Expenses incurred when adding new assets to the existing production line
$Cost_{Installation_a}$	Installation cost for asset a
$Cost_{Installation_i}$	Installation cost for asset i
$Cost_{Purchase}$	Expenses related to buying new assets necessary to meet new customer demands
$Cost_{Purchase_a}$	Purchase cost for asset a
$Cost_{Purchase_k}$	Purchase cost for asset k
$Cost_{Reconfiguration}$	Total reconfiguration cost, constituting the summation of installation, purchase, and removal costs
$Cost_{Removal}$	Expenses associated with removing obsolete or unnecessary assets from the production line
$Cost_{Removal_a}$	Removal cost for asset a
$Cost_{Removal_j}$	Removal cost for asset j
d	Index representing a specific candidate asset, where $d \in [1, k_m] \cap \mathbb{Z}$
$f(\boldsymbol{\theta_a})$	Function denoting the forward kinematics of the robot, mapping the joint angles to Cartesian coordinates
h	Total number of sub-evaluators
i	Index representing a specific sub-evaluator, where $i \in [1, h] \cap \mathbb{Z}$
$J(\boldsymbol{\theta_a})$	Jacobian matrix of the robot at joint angles θ_a

$J(\theta)$	Jacobian matrix of the robot at joint angles θ
k_m	Total number of candidate assets considered for process m, where $k_m \in [k_1, k_j] \cap \mathbb{Z}$
m	Index representing a specific process, where $m \in [1, j] \cap \mathbb{Z}$
n	Total number of combinations of the candidate assets for Product II
O	Original asset list
P_a	Coordinates of the end effector pose at the operation point a
q	Quaternion representing orientation
q_0, q_1, q_2, q_3	Components of quaternion vector
R_i	Consistent fuzzy matrix for sub-evaluator i
R_1	Resultant or related matrix derived from B_1 in a fuzzy logic context
r_{ef}^i	Consistent fuzzy matrix element representing relationship strength between D_{me} and D_{mf} under sub-evaluator U_i
S_d^i	Score of the candidate asset under sub-evaluator U_i indicating matching ability
SA	Set of all assets
SA_1	Set of assets for Workstation 1
SA_2	Set of assets for Workstation 2
s_d^1	Score or value assigned to the first option or criterion for decision d
s_d^2	Score or value assigned to the second option or criterion for decision d
t_{cycle}	Total time for a robot to complete one operational cycle
$t_{movement}$	Movement time component of the cycle time
$t_{operation}$	Operation time component of the cycle time
t_{wait}	Waiting time component of the cycle time
W_{imd}	Weight of the d-th candidate asset for sub-evaluator i in process m
W_{md}	Set of updated weights for asset numbered d in process m, represented as a collection across all sub-evaluators
W_{total}	Total robot manoeuvrability index of the entire robotic assembly unit
w	Absolute value of determinant of Jacobian matrix
w_a	Absolute value of determinant of Jacobian matrix at operation point a
X	End effector pose of the robot
x_c, y_c, z_c	Coordinates of the center point of the workpiece
θ_a	Joint angle vector for point a

ϕ	Roll angle in orientation representation
ψ	Yaw angle in orientation representation
θ	Pitch angle in orientation representation
TSES	Total specification efficiency score representing overall efficiency in a selection or assessment context
✓	Symbol indicating validation, confirmation, or success in a particular criterion or condition

Chapter 1

Introduction

1.1 Background and Motivation

Digital manufacturing, also known as Industry 4.0, brings in a new era for production. The integration of cutting-edge technologies into the industrial ecosystem, including the Internet of Things (IoT), big data analytics, artificial intelligence, and state-of-the-art robotics [1, 2, 3]. Fundamentally, Industry 4.0 makes it possible to collect and analyse data in real time, thereby improving and accelerating industrial operations.

The increased efficiency and flexibility in design and production processes, which result in significant cost savings, is a direct result of this digital revolution [4]. For instance, advanced analytics can analyse manufacturing data to identify inefficiencies and facilitate prompt interventions. Additionally, virtual prototyping and testing tools are available in technology to shorten design cycles and reduce dependency on physical prototypes.

A salient feature of this digital era is the active involvement of the end-user in the manufacturing process. Advanced digital platforms empower customers to mould products according to their requirements, ushering in an age of mass customisation. This shift is underpinned by technological innovations, notably reconfigurable manufacturing systems (RMSs) [5], that facilitate seamless configuration modifications. The recent COVID-19 pandemic underscored the indispensability of such adaptable manufacturing systems. For example, during the pandemic, the Office for National Statistics (ONS) data for February 2021 suggests that monthly man-

ufacturing grew by 1.3% but remains 4.2% below its February 2020 level. A UK-wide survey by Make UK suggests that 52% of manufacturing firms had to make redundancies due to the pandemic, and there was a 57% drop in apprenticeships around manufacturing technologies. Approximately 80% of manufacturing firms used furlough to some extent, highlighting the reliance on government support to sustain operations. Over the past 12 months, manufacturing output contracted by 23.1%. Major automotive manufacturers like Nissan and Jaguar Land Rover faced significant challenges, with all major car plants closing at the start of the pandemic and some remaining closed for months [6].

These challenges highlight the potential benefits of RMSs in such crisis situations. The RMS, by design, offers flexibility and adaptability in manufacturing operations. They could have enabled manufacturers to rapidly reconfigure production lines to meet changing demands, such as the shift to producing essential items like personal protective equipment (PPE) or medical devices during the pandemic. This flexibility might have mitigated some of the negative impacts, like the reduction in manufacturing output and the need for redundancies. Furthermore, the adoption of RMS could have facilitated quicker resumption of operations post-lockdown by allowing manufacturers to implement social distancing measures on the factory floor efficiently. Essentially, RMS could have served as a critical tool in enabling manufacturers to maintain production levels and adapt to new market demands amidst the unprecedented challenges posed by the pandemic [6].

Robots are crucial in modern manufacturing, and their importance in RMS cannot be overstated. In the realm of manufacturing science and technology, the evolution of robotic assembly systems within RMSs brings forth distinct challenges. Scientifically, a primary challenge is the efficient processing and integration of heterogeneous data originating from the diverse systems and technologies intrinsic to modern robotic assembly cells [7]. Achieving a unified understanding from this varied data is critical for real-time decision-making and requires sophisticated models that can handle the complexity and variability inherent in manufacturing data. From a technological perspective, this translates into developing robust data processing frameworks that can seamlessly integrate and interpret information from varied sources, ensuring that robotic assembly cells can rapidly respond to changes with minimal latency. This requires not only advanced algorithms for data analysis and interpretation but also innovative approaches to data architecture and management.

Another significant challenge lies in the consistent adaptability of these systems to the ever-changing consumer landscape, reflecting the market's dynamic demands [8]. Scientifically, this involves understanding and predicting how changes in process requirements will affect the system's performance and devising strategies to enable quick and efficient reconfiguration. This is particularly challenging in a market characterised by rapid and often unpredictable changes. Technologically, enabling such adaptability involves creating modular and flexible systems that can be quickly reconfigured without extensive downtime.

Lastly, achieving optimal operational efficiency through continuous optimisation, especially concerning layout and post-assembly processes, is a multifaceted challenge [5]. Scientifically, it requires a deep understanding of the factors that contribute to operational efficiency and the development of models that can identify opportunities for optimisation in a constantly changing environment. Technologically, this involves implementing sophisticated optimisation algorithms that can work in real time, making quick adjustments that improve efficiency without disrupting ongoing operations. It also involves designing flexible systems to be reconfigured or adjusted on the fly as new information becomes available or conditions change.

Addressing these challenges necessitates a comprehensive and multidisciplinary approach, merging insights from various fields to enhance the adaptability and efficiency of robotic assembly cells in contemporary manufacturing. By tackling these issues, the research sets a pathway towards more responsive, flexible, and efficient manufacturing systems capable of adapting to the fast-paced and ever-evolving market demands.

1.2 Aim and Objectives

1.2.1 Research Questions

Based on these challenges, this research seeks to answer the following questions:

1. Research Question 1 (RQ1)
   How can data from various sources be efficiently processed by integrating diverse systems and technologies into robotic assembly cells?
2. Research Question 2 (RQ2)

How can robotic assembly cells adeptly adjust to ever-shifting process requirements, reflecting the changing consumer market?

3. Research Question 3 (RQ3)
   How can a reconfigurable robotic assembly system efficiently optimise its operations, especially in layout, after adapting to current process requirements?

1.2.2 Aim

This PhD research aims to develop a framework for reconfiguring robotic assembly cells, focusing on the efficient processing of heterogeneous manufacturing data, adaptation to changing product requirements, and post-adaptation layout optimisation, with validation through software development and use cases.

1.2.3 Objectives

The primary research aim is supported by the following objectives, which are aimed at addressing the research questions:

- Objective 1: To develop an integrative approach for effectively managing heterogeneous manufacturing data within robotic assembly cells, which addresses RQ1 and supports informed decision-making.
- Objective 2: To design a reconfigurable robotic assembly cell system that exhibits agility and responsiveness to changing market demands and product requirements, thereby answering RQ2.
- Objective 3: To formulate a post-adaptation optimisation process for robotic assembly cells, focusing on layout optimisation using artificial intelligence, knowledge graphs, and simulation methodologies, in response to RQ3.
- Objective 4: To validate the strategies of the preceding three objectives through software development and testing within use cases.

These objectives, while addressing the respective research questions, collectively underpin the central aim of the research: to develop a framework for the reconfiguration of robotic assembly cells that meets the challenges of the robotic assembly reconfiguration cell and validate it through software development and use case testing. These challenges include, for example, the need to process and integrate heterogeneous manufacturing data efficiently, the requirement for rapid adaptation to changing market demands and product specifications, and the imperative for continuous optimisation of the system layout to maintain operational efficiency and productivity. Each of these challenges represents a significant hurdle in the path to creating more flexible, efficient, and responsive manufacturing systems, and the proposed framework aims to provide robust solutions to these issues.

1.3 Thesis Structure

The structure of this thesis is shown in Figure 1.1.

In more detail, the contents per chapter are as follows:

- **Chapter 1:** This chapter describes the background, motivation, research aims, research questions and objectives of this PhD thesis.
- **Chapter 2:** This chapter offers an extensive literature review on critical concepts and technologies pertinent to the proposed methodology. Topics covered include a comprehensive examination of RMSs, ontology modelling in the manufacturing domain, knowledge graphs, asset selection, layout optimisation, simulation software, and multi-objective optimisation. Additionally, it delves into the limitations and gaps in existing approaches, setting the context for the need for the proposed methodology.
- **Chapter 3:** This chapter offers an overview of the proposed methodology concerning reconfiguration information within the robotic assembly process. It outlines the primary contributions of the PhD thesis, emphasising their interrelation and significance. Moreover, criteria set out for validation are introduced to ensure the methodology's robustness and relevance.
- **Chapter 4:** This chapter addresses RQ1 and also contributes to fulfilling Objective 1 of

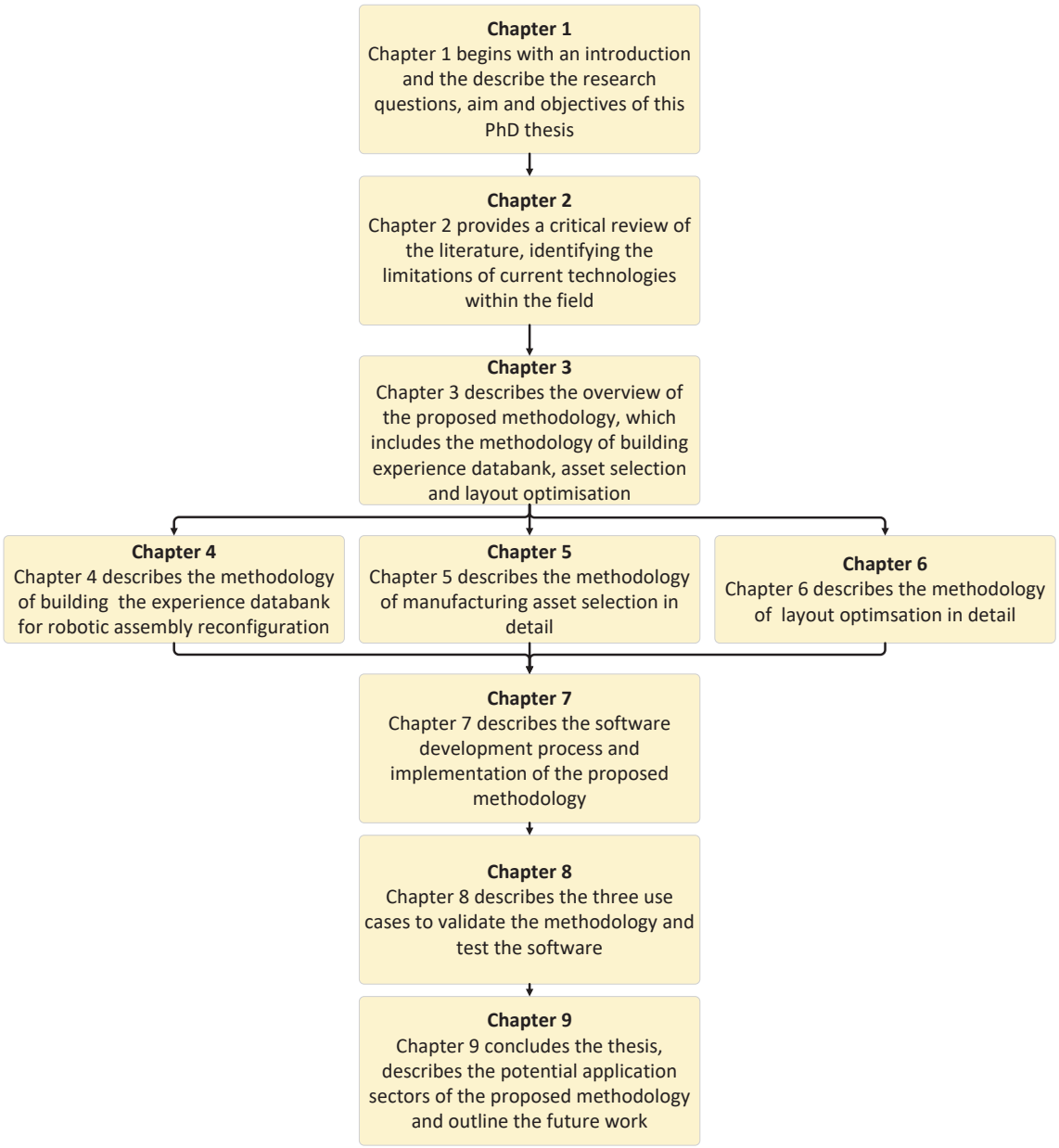


Figure 1.1: Structure of this PhD thesis

the PhD thesis. It delves into an ontology model tailored for robotic assembly reconfiguration and elaborates on its application in knowledge graphs. The chapter introduces the ontology model and delineates the construction of the experience databank via the knowledge graph, utilising the ontology model.

- **Chapter 5:** This chapter addresses RQ2 and concurrently aims to meet Objective 2 of the PhD thesis. It delves into the methodology for asset selection, detailing the criteria and objectives used to pinpoint candidate assets and to determine the most suitable assets for specific manufacturing processes.

- **Chapter 6:** In addressing RQ3 and meeting Objective 3, this chapter focuses on the development of a modular methodology for layout configuration optimisation, characterised by its multi-objective and scalable nature.

- **Chapter 7:** This chapter marks the shift from theory to practice, highlighting the software implementation aspect of the RMS methodology, which is a key component of Objective 4. It outlines the development processes for three core components: the experience databank interface, the asset selection tool, and the layout optimisation solution. The progression from design to refinement is detailed, demonstrating the transformation of abstract concepts into tangible software tools for contemporary manufacturing.

- **Chapter 8:** This chapter evaluates the proposed methodology through three distinct use cases to fulfil the remaining aspects of Objective 4. The first use case tests asset selection and system layout optimisation using the experience databank. The second assesses layout optimisation for a single robot with multi-criteria, and the third investigates modular asset selection with multiple criteria, both utilising the experience databank. These cases collectively validate the methodology's adaptability and efficiency in diverse manufacturing contexts.

- **Chapter 9:** This chapter provides a holistic summary of the research's pivotal contributions within the realm of RMSs. It candidly addresses the study's limitations, offering a balanced perspective on its findings. Moreover, identifying potential research avenues sets the stage for future explorations in this dynamic field.

Chapter 2

Literature Review

This chapter presents a comprehensive literature review, beginning with the exploration of RMS and their integration into robotics. The structure of this chapter is depicted in Figure 2.1.

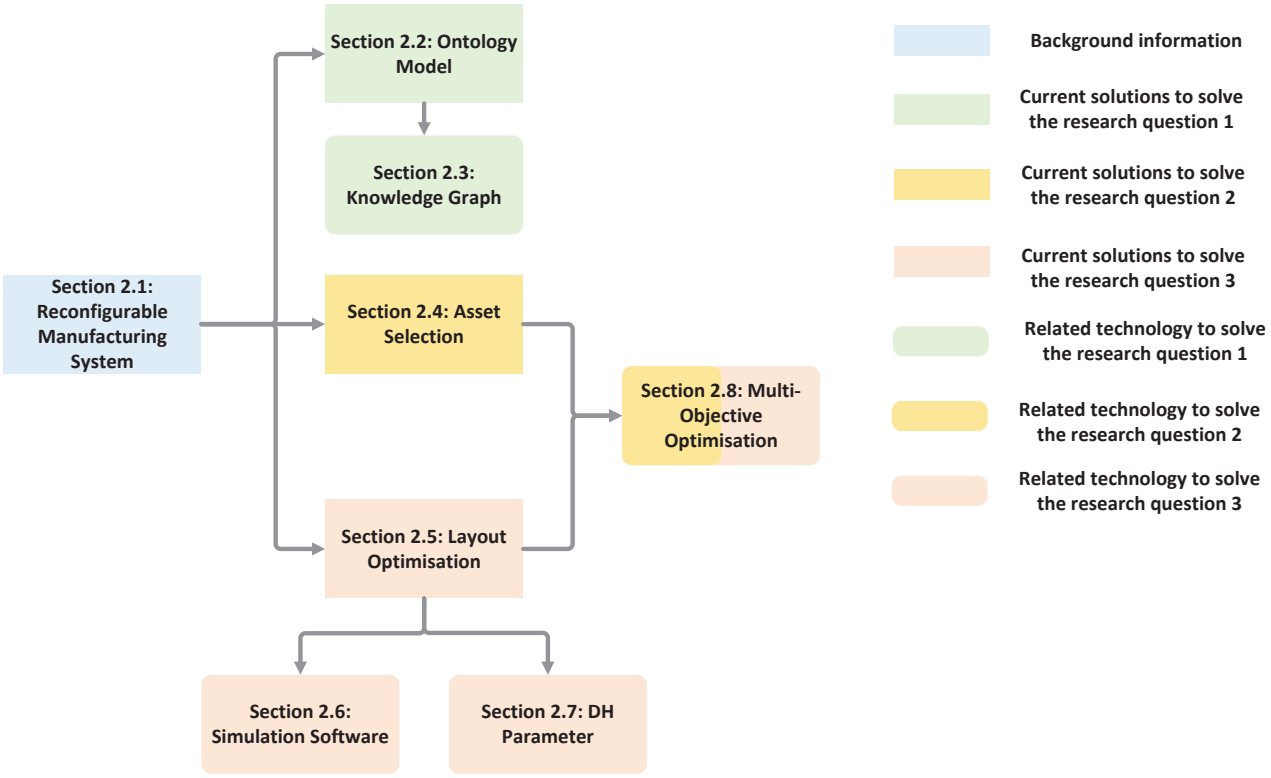


Figure 2.1: Structure of the literature review chapter

The challenges associated with implementing RMS in robotics are detailed in Section 2.1. This sets the stage for a deeper investigation into the various research questions posed in this thesis.

Section 2.2 addresses the RQ1, focusing on proficient data management within robotic assembly cells. It delves into existing knowledge representation techniques, particularly ontology models, and discusses their limitations and applications in the manufacturing domain.

Building upon the foundational concepts of ontology models, Section 2.3 introduces knowledge graphs to enrich the representation and understanding of complex data structures. This section elucidates how knowledge graphs can be leveraged to enhance data management and interpretation in manufacturing systems.

In Section 2.4, attention shifts to the RQ2, which scrutinises the current methodologies employed in asset selection. This section outlines the prevailing techniques and their inherent limitations, setting the stage for potential improvements.

To address the RQ3, Section 2.5 reviews the existing solutions for layout optimisation in manufacturing systems. It critically assesses these solutions and highlights their limitations, paving the way for the development of more efficient and effective optimisation strategies.

Section 2.6 discusses the role of simulation software in facilitating layout optimisation. It reviews the current simulation tools used in manufacturing and outlines the criteria for selecting appropriate software for this thesis.

The Denavit-Hartenberg (DH) parameters, fundamental for determining robot manoeuvrability, are explored in Section 2.7. These parameters are crucial for the optimisation objectives of layout configuration optimisation discussed in this thesis.

Finally, Section 2.8 introduces the concept of multi-objective optimisation, which is employed for both asset selection and layout optimisation. This section sets the stage for the subsequent chapters, which will delve into the application of these concepts in addressing the research questions.

2.1 Reconfigurable Manufacturing Systems and Robotic Assembly

2.1.1 Reconfigurable Manufacturing Systems

RMS embodies the vision of a flexible and adaptive manufacturing landscape. By definition, an RMS is designed with a quick-change capability, ensuring that it can be reconfigured in real time to address dynamic production requirements. As manufacturing paradigms shift towards more customer-centric models, the systems that can rapidly adjust to new production needs have become of paramount importance.

The idea behind RMS arose from the limitations of traditional manufacturing setups, which were either too rigid (dedicated systems) or too flexible (flexible manufacturing systems) but with high retooling times. An RMS, as proposed by Koren et al. [5], aims to strike a balance by offering the speed of dedicated systems and the versatility of flexible systems.

However, while the concept of an RMS is promising, its real-world implementation is not without challenges. A primary challenge is the seamless integration of various modular components, ensuring that the system can genuinely respond in real time to production shifts. Furthermore, as Morgen et al. [9] noted, while the modularity of RMSs offers advantages in terms of adaptability, it also presents complexities in system planning, especially when predicting future production needs.

Another significant aspect is the cost associated with RMSs. While such systems should, theoretically, lead to cost savings in the long run due to their adaptability, the initial investment can be substantial [10].

In summary, while RMSs offer a transformative approach to modern manufacturing, ensuring their seamless integration, scalability, and cost-effectiveness remains an active area of research.

2.1.2 Robotic Assembly in Reconfigurable Manufacturing Systems

In the dynamic realm of modern manufacturing, integrating robotics into RMSs offers adaptability and efficiency [11]. Robots, built with sophisticated reconfigurability features, have

emerged as vital cogs in the machinery, enabling manufacturers to adapt swiftly to changing production needs without undergoing extensive overhauls. Given fluctuating market demands and rapid technological shifts, this adaptability is not a mere luxury but a necessity.

The evolution of robots in RMS highlights the significant leaps in technological advancements. Enhanced sensors give these robots a heightened sense of their environment, leading to more precise operations. Merging machine learning and artificial intelligence has taken these robots a step further than being mere programmable entities: they are now adaptive systems capable of learning and evolving in real time. As Lee et al. [12] highlighted, the reprogramming capabilities of these robots allow for adaptability, thus ensuring minimal downtime, which, in turn, maximises production efficiency.

However, the evolution of robotic assembly systems has inherent challenges. Key among them is the task of processing data that originates from many sources. This challenge arises due to the integration of diverse systems and technologies intrinsic to modern robotic assembly cells [7]. The heterogeneity of the resulting data can create difficulties in achieving a unified understanding, which is essential for instantaneous decision-making.

Additionally, the ever-changing consumer landscape requires the consistent adaptability of these systems to diverse process requirements [8]. The market's demands are in constant flux, evolving, shifting, and, at times, undergoing rapid changes.

Lastly, the pursuit of optimal operational efficiency requires continuous tweaks, especially concerning layout optimisation processes [5]. It is also difficult to find a scalable optimisation framework to do the layout optimisation.

2.2 Ontology Models in the Manufacturing Domain

To answer the RQ1, emphasis on proficient data management within robotic assembly cells becomes key. This underscores the need for an in-depth understanding of knowledge representation within the manufacturing realm. Recent trends in the manufacturing sector reflect amplified engagement with knowledge representation technologies. Specifically, ontologies [13, 14], semantics [15, 16], and semantic web technologies [17, 18] have been increasingly adopted. These tools not only provide collaboration and interoperability but also offer adaptability —

essential attributes for addressing the changing intricacies of the manufacturing landscape.

Historically, one of the pioneering manufacturing ontologies was the Process Specification Language (PSL), developed to provide a neutral language for representing process-related knowledge, thereby supporting application integration [19]. However, while Extensible Markup Language (XML)-based methodologies primarily addressed the structural description of manufacturing processes, they often overlooked the implicit semantic content, making them less versatile [20].

In more recent advancements, Lu et al. [21] introduced an ontology-centric approach, envisioning the enhancement of semantic interoperability throughout the service provision in the cloud. Similarly, Wang et al. [22] proposed an ontology model to describe task semantics in cloud manufacturing systems. This model, however, lacks a vendor-neutral resource description, posing challenges for interoperability. Ontologies play a crucial role in facilitating proficient data management within robotic assembly cells, primarily due to their capabilities in knowledge representation. However, a significant gap emerges: the practical application and potential of the method in complex decision-making scenarios remain vaguely defined.

Järvenpää et al. made a noteworthy contribution to developing the MaRCO information model [14]. This model offers resource vendors a standard, vendor-independent descriptor for resource capabilities, proving instrumental in capability matching. Its primary utility lies in assessing the suitability of a manufacturing process or equipment relative to specific production requirements. However, its current scope of application remains constrained due to a lack of widespread adoption in the industry. Moreover, its inability to consider dynamic parameters or reconfiguration nuances is a limitation. My proposed methodology in this PhD thesis overcomes these.

While the literature offers various models and methodologies, persistent challenges and gaps underscore the need for further research:

- Many prevailing resource description methods are domain-specific, offering solutions tailored to particular applications and lacking a broader, more holistic perspective.
- Several models either lack practical applications or neglect the necessity of vendor-neutral descriptions [21, 22].

- The development of models that can incorporate static and dynamic parameters is essential.
- A universally accepted industry standard or framework for ontology models in manufacturing remains elusive.

In conclusion, considerable progress has been achieved in the field of ontology models within manufacturing. Nonetheless, tackling the highlighted gaps, especially the creation of comprehensive, dynamic, and vendor-neutral models, is crucial for the onward direction of ontology applications in the sector.

2.3 Knowledge Graph

Building upon the foundational concepts established by ontology models in the manufacturing domain, knowledge graphs further enrich the understanding and representation of complex data structures. These graphs, with their intricate connections, serve as an extension and evolution of the principles of ontology models.

In 2012, Google introduced its Knowledge Graph project, which the company leveraged to improve the relevance of search results and enrich the user experience. This announcement spurred a wave of developments, with a plethora of knowledge graphs emerging, propelled by the increase in online resources and the advent of linked open data (LOD) initiatives. The application spectrum of knowledge graphs is vast, encompassing domains such as question-answering systems [23, 24] and recommendation systems [25, 26, 27], to information retrieval [28, 29, 30]. Knowledge graphs methodically structure data about the real world, delineating entities and their extensive networks of relations [31]. Within these graphs, data is articulated as a triple (h, r, t), which signifies that there is a relationship r connecting a head entity h to a tail entity t, exemplified by *(Milling machine, hasTools, Milling cutter)*. Diverging from traditional data storage, which is often static and not well-suited for dynamic updates, knowledge graphs are predicated on the accumulation, governance, and interpretation of unstructured, heterogeneous data. This architecture enables them to dynamically adjust to new data and relations, rendering them exceptionally pertinent for perpetually evolving sectors such as manufacturing.

The inherent capacity of knowledge graphs to delineate and process interconnections is a fundamental advantage, facilitating intricate associative analyses and the deduction of novel insights [32]. Knowledge graphs possess the flexibility to incorporate additional entities and relations without necessitating alterations to their foundational schema. This makes them particularly effective for complex query resolution involving numerous entities and their interrelations [33]. For instance, answering a question such as "How to produce the fuselage of an aircraft?" would pose a challenge for conventional relational databases or flat file structures.

A knowledge graph is structured into two primary layers: the schema layer and the entity layer [34]. The former encompasses the abstract concepts, attributes, and the interplay between these concepts. The latter layer is populated with distinct entities derived from the schema layer's conceptual framework. For example, the relationship *(Milling machine, hasTools, Milling cutter)* illustrates an entity relationship within the entity layer, where both the milling machine and milling cutter are instances of the "Asset" category from the schema layer, connected by the semantic relationship *(Asset, hasTools, Asset)*. Knowledge graphs' data structures are well-aligned with the underlying frameworks used in various artificial intelligence (AI) applications, such as those dealing with diverse, multi-relational big data. This alignment offers robust support for advanced applications in intelligent searching, interactive Q&A systems, smart recommendation engines, and comprehensive data analytics [35]. Knowledge graphs can be housed in two predominant storage systems: Resource Description Framework (RDF)-based systems and graph databases. The former is adept at facilitating data interchange and distribution, while the latter excels in executing efficient graph-based queries and exploration. Furthermore, RDF storage systems manage data as triples without encompassing attributes, whereas graph databases typically employ attribute graphs, enabling them to embed attributes within entities and relationships, thus providing a more nuanced representation of complex business scenarios [36].

2.3.1 Applications in Manufacturing

Recently, knowledge graphs have been used in the industrial industry. Zhou et al. [37] introduced a unified knowledge graph-driven production resource allocation approach that facilitated quick decisions on resource allocation for a specified task order, taking into account device assessment strategy and resource machining information. An industrial knowledge graph-based

multi-agent reinforcement learning technique for creating a Self-X cognitive manufacturing network was presented by Xia et al. [38]. The relationships between entities can be expressed effectively using knowledge graphs. They enable individuals to analyse issues using the relationships between knowledge. Despite the numerous benefits of knowledge graphs in manufacturing, some limitations must be considered, including the following:

- **Data Integration**: Integrating data from multiple sources can be challenging and time-consuming. The quality and format of the data may also vary, making it difficult to standardise and integrate the information into a knowledge graph [39].
- **Expertise**: Building and maintaining a knowledge graph requires specialised skills and expertise, making it challenging for organisations to implement one themselves [40].
- **Lack of Standardisation**: The lack of standardisation in knowledge graph technology and data representation can limit interoperability and integration with other systems [41].

2.3.2 Tools to Implement a Knowledge Graph

The generation of knowledge graphs, premised on ontology models, has been significantly advanced by a variety of methods and software tools. These encompass both open-source platforms and proprietary solutions, each offering distinct capabilities:

- **Neo4j:** This open-source graph database management system enables efficient storage, querying, and visualisation of graph-structured data. Its powerful Cypher query language facilitates intricate and effective graph operations and navigations between node relationships [42].
- **Protégé:** Protégé is an open-source ontology editor and knowledge-based framework that supports the creation, editing, and visualisation of ontology models. It accommodates several ontology languages, including Web Ontology Language (OWL), and RDF, and RDF Schema (RDFS), and it provides a comprehensive set of plugins for ontology reasoning, querying, and visualisation [43].
- **Apache Jena:** This open-source Java-based framework is designed for building Semantic Web and Linked Data applications. It delivers Application Programming Interfaces

(APIs) for reading, writing, and querying RDF data, along with support for reasoning with OWL ontologies. A SPARQL query engine is included for executing complex queries over RDF data [44].

- **GraphDB:** GraphDB manages semantic information effectively with on-the-fly data transformations. It offers robust consistency, reasoning, and semantic similarity search algorithms. Furthermore, its user-friendly cluster management interface and semantic graph visualisation tools make it an appealing choice [45].

- **Stardog:** Stardog is an enterprise graph database platform offering data unification, data integrity, and data discovery. It supports RDF and SPARQL standards, and it can manage vast volumes of triples in a graph while providing advanced security features and high availability [46].

- **Blazegraph:** Blazegraph is an ultra-high-performance graph database supporting the RDF data model. It is well-suited to large-scale RDF data sets and offers optional support for quad-store and provenance models. Its graphics processing unit (GPU) acceleration makes it capable of managing exceptionally large graphs [47].

- **RDFlib:** RDFlib is a Python library devised for working with RDF data structures. It encompasses parsers and serialisers for a plethora of RDF formats, including RDF/XML, Notation3, NTriples, N-Quads, Turtle, and TriX, and it furnishes a graph interface underpinned by various storage back-ends [48].

- **Virtuoso:** Virtuoso is a high-performance and scalable multi-model relational database management system (RDBMS), serving as data integration middleware, a linked data deployment facility, and an Hypertext Transfer Protocol (HTTP) application server platform. It is distinguished for its capacity to expedite the transformation of basic business data into valuable insights [49].

- **Grakn:** Grakn is a knowledge graph database to manage complex data. It utilises a schema to model data that incorporates the inherent structure of the domain. Its query language, Graql, allows users to define, query intuitively, and reason over the knowledge graph [50].

- **Py2neo:** Serving as both a client library and a toolkit, Py2neo facilitates interaction with the Neo4j database from Python-based applications as well as via the command line.

This library is compatible with Bolt and HTTP, offering a comprehensive Application Programming Interface (API) for higher-level operations. Additionally, it encompasses administrative tools, an interactive shell, and a dedicated Cypher lexer compatible with Pygments, alongside a suite of other advanced features [51].

In summary, the comparison of various knowledge graph implementation tools is concisely presented in Table 2.1. This table offers a clear and structured overview of each tool, detailing its unique capabilities and potential limitations, thus providing valuable insights for those seeking to implement knowledge graphs in their respective domains, particularly in manufacturing. This comprehensive comparison serves as a useful guide for selecting the most appropriate tool based on specific project requirements and constraints.

Table 2.1: Comparison of Knowledge Graph Implementation Tools

Tool	**Capabilities**	**Limitations**
Neo4j	Efficient graph operations, Cypher query language, suitable for large datasets	Steep learning curve for Cypher language
Protégé	Supports multiple ontology languages	Complex for beginners, plugin dependency
Apache Jena	Semantic Web and Linked Data applications	Java-based, limited to Java developers
GraphDB	Semantic information management	Complex setup, cost for enterprise features
Stardog	Data unification, RDF/SPARQL support	Cost, specialised knowledge needed
Blazegraph	High-performance, GPU acceleration	Limited community support
RDFlib	Python-based, flexible storage back-ends	Limited to Python, requires programming knowledge
Virtuoso	Multi-model database, data integration	Complex for non-experts
Grakn	Schema-based modelling, Graql query language	Limited adoption
Py2neo	Toolkit for Neo4j, Python compatibility	Specific to Neo4j and Python

2.4 Asset Selection in Robotic Assembly Cells

The RQ2 concerns the main problem in the complex world of production, namely guaranteeing that robotic assembly cells can quickly adapt to constantly changing process requirements. Asset selection, a procedure long recognised as a cornerstone of effective production systems, is inextricably tied to this problem. As production environments evolve due to technological

advancements and changing consumer preferences, the significance of selecting the right assets, whether machines, robots, or other essential tools, becomes paramount. Such decisions historically relied significantly on professional intuition, accumulated experience, and fundamental data analysis. However, the digital revolution in the industrial industry has triggered a move towards more advanced, data-driven processes.

Contemporary asset selection methodologies in the broader manufacturing context lean heavily on predictive analytics. The emergence of Industry 4.0 has underscored the importance of harnessing data for smarter decision-making [52]. Predictive models, particularly those underpinned by machine learning algorithms, have provided game-changing insights into asset performance, life-cycle costs, and potential bottlenecks [8]. Tools such as the "Analytic Hierarchy Process (AHP)" [53] and "Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS)" [54] have become indispensable. These frameworks allow manufacturers to evaluate and rank assets systematically based on multifaceted criteria, thereby optimising performance and costs.

The stakes and complexities of asset selection are further heightened in robotic assembly. Robots are multifunctional entities that can perform various tasks, from simple pick-and-place operations to intricate assembly routines. As highlighted by Kadir et al. [55], selecting the right robotic assets in assembly cells is not only about performance but also about flexibility, adaptability, and integration with other systems. The challenge is compounded by the sheer variety of robotic assets available today, each with unique capabilities, limitations, and cost implications.

The existing approaches, while powerful, have certain limitations. For instance, while the models offer insights based on historical data, they sometimes struggle with scenarios that deviate from past patterns [56]. The dynamic nature of an RMS, with its unpredictable and ever-changing environment, often results in such deviations. This calls for more adaptive and real-time decision-making frameworks that swiftly recalibrate based on new data and emerging scenarios.

Moreover, as noted by Bi et al. [57], there is a critical need for comprehensive frameworks that consider both the individual performance of assets and their synergy when incorporated into an assembly cell. This synergy can be quantified through methodologies such as performance metrics analysis, which includes the concept of cost-efficiency. For instance, when different robotic

assets are combined, the total cost of operation may decrease due to improved efficiency and reduced redundancy. This cost synergy arises not just from the individual assets' capabilities but also from how they complement each other in operation, leading to a more cost-effective production process. The interdependencies between robotic equipment and other industrial system components significantly impact output and overall effectiveness. Manufacturers can make more informed decisions by evaluating both the individual and collective cost implications alongside other performance metrics. This integrated approach, assessing individual asset performance in conjunction with their combined cost and operational efficiencies, is crucial in optimising the overall system efficiency. It also ensures adaptability and flexibility within the dynamic environment of reconfigurable manufacturing systems. In summary, gaps remain despite noteworthy developments in asset selection for robotic assembly cells. The ever-evolving nature of RMS demands continuous innovation in asset selection methodologies to ensure that these systems remain relevant, adaptive, and holistic.

2.5 Layout Optimisation in Robotic Assembly Cells

Understanding the complexities of layout optimisation in robotic assembly systems is essential for answering RQ3, which asks how the need for more optimisation, especially regarding layout and operating efficiency, can be determined post-assembly cell changes. Layout optimisation is crucial in the present dynamic industrial scene. Adaptability is needed throughout a business as market demands change quickly. This calls for more streamlined material flows, improved inventory efficiency, and significant cost savings in material handling. Beyond these quantifiable indicators, the broader goal is to ensure flexibility, safety, and the overall productivity of the assembly process.

Historically, layout optimisation heavily relied on heuristic strategies, steered predominantly by the expertise of plant engineers [58, 59]. However, as the complexities of modern manufacturing burgeoned, the limitations of these traditional human-centric methods became evident.

With the development of computational algorithms, evolutionary algorithms and machine learning algorithms have recently been used in layout optimisation. Caputo et al. [60] presented a method based on a genetic algorithm for optimising a process plant layout. Klar et al. [61] proposed a framework for automated multi-objective factory layout planning using

reinforcement learning. Souilah [62] proposed a methodology to group resources into manufacturing cells, design the intra-cell layout, and place the manufacturing cells on the available shop-floor surface.

Yet, the algorithms are not without their gaps:

- Although powerful, many algorithms require vast and diverse datasets for training, which poses a challenge in rapidly evolving manufacturing sectors [63].
- Many algorithms, while adept at handling specific optimisation scenarios, struggle when faced with multi-objective problems where goals might conflict [64].

When examining the scope of layout optimisation, contemporary research predominantly focuses on two distinct levels: the machine and the system. At the machine level, studies focus primarily on the modification of individual machine configurations. Such studies aim to optimise specific units within a larger system, as highlighted by Liu et al. [65]. Conversely, system-level studies adopt a broader perspective: they emphasise both machine selection and their subsequent spatial configuration within the layout. The objective is to strategically position chosen machines within a predefined area, ensuring optimal operational flow and efficiency. Seminal works in this domain include studies by Haddou et al. [66] and Ghanei et al. [67].

However, these focused approaches present evident limitations:

- The lack of integration between machine-level and system-level optimisation constitutes a significant gap in current methodologies. This siloed approach potentially overlooks the synergistic benefits that a combined perspective might offer.
- Most current research mainly focuses on the machining of parts. Other areas of the industry, which might present new opportunities, do not receive as much attention, as noted by Yelles-Chaouche [68].

2.6 Simulation Software for Robotic Assembly Cells

In the Industry 4.0 landscape, simulation tools have become integral to the functioning of robotic manufacturing. These tools, which echo the digital twin concept, provide a crucial

platform for analysis, optimisation, and testing within a virtual environment that accurately reflects real-world setups. They offer a risk-free milieu for experiments and validations, which is particularly valuable given the growing complexities in industrial automation and the need for dependable and flexible simulation solutions.

The field of industrial robotics is replete with a variety of simulation methodologies, ranging from all-encompassing commercial software suites to bespoke, task-specific Python libraries. Each of these comes with its own set of features and foundational principles. The examination of these simulation tools is not merely for theoretical enrichment; it is essential for tackling the RQ3, which zeroes in on the critical need for post-assembly cell adjustments and the push for ongoing layout optimisation. Significantly, these simulation resources are indispensable for refining layout designs, allowing for continuous iteration and evaluation without the need for physical alterations or incurring real-world costs.

Siemens Tecnomatix Process Simulate [69, 70] stands out for its exhaustive functionality that supports an entire manufacturing ecosystem. As indicated in the literature, it is especially adept at handling intricate systems involving multiple machines at the system level.

On the other hand, when focusing on robot-centric applications, platforms such as ABB RobotStudio [71, 72], FANUC Roboguide [73, 74], and RoboDK [75] are notable contenders. While RobotStudio and Roboguide are proficient in their domain, they are vendor-specific and optimised for use with their own brand's equipment. This specialisation ensures seamless integration but could restrict versatility when dealing with robots from various manufacturers. RoboDK, however, stands out for its brand-agnostic approach, enabling quick simulation of a wide range of robots from different vendors.

Siemens Tecnomatix Process Simulate is noted for its extensive capabilities, catering to a comprehensive manufacturing ecosystem. As referenced by Hovanec [76], it is particularly suitable for complex setups involving multiple machines at a system level. However, when considering robot-centric software, ABB RobotStudio, FANUC Roboguide, and RoboDK emerge as prominent choices. As highlighted by Neto [77] and Bulej [78], RobotStudio and Roboguide, while proficient, are vendor-specific platforms, optimised primarily for their respective brands. This specificity ensures impeccable integration but may offer limited flexibility across multiple brands.

Conversely, RoboDK, as cited by Chakraborty [79], offers versatility. It isn't anchored to a specific robot brand, allowing for the rapid simulation of various robots from different manufacturers. Its capability to efficiently handle individual robot simulations, coupled with a straightforward approach, makes it an apt choice for projects emphasising multi-brand flexibility and a single robot focus.

Open-source Python tools, such as PyBullet [80], have brought about a new level of customisation in technology. These tools are highly adaptable and can be tailored to users' specific requirements. However, utilising them in complex scenarios or integrating them with existing systems can be time-consuming and requires expertise.

The variety of simulation tools for industrial robotics caters to diverse requirements. The appropriate choice depends on the intricate details and complexities of the simulation objective. This study adopts a simulation-centric methodology to explore RQ3 concerning the framework for layout optimisation.

2.7 Denavit-Hartenberg Parameters

The DH parameters, attributed to Jacques Denavit and Richard S. Hartenberg, provide a structured and efficient means of defining the geometry of a robot manipulator. First introduced in a foundational 1955 paper, the DH parameters have since become a standard approach in robotics engineering [81]. In this PhD thesis, these parameters are fundamental for determining robot manoeuvrability, which is used as one of the optimisation objectives of the layout configuration optimisation.

The DH parameters offer a standardised method to define and connect coordinate frames for each joint in a robotic system, which helps to simplify the kinematic equations governing the movement and operation of the robot. The fundamental idea is to attach a coordinate system to each joint of the robot and then define the transformations between these coordinate systems using only four parameters:

- Link length (a) - the distance between two consecutive joint axes.
- Link twist (α) - the angle between two consecutive joint axes.

- Link offset (d) - the perpendicular distance from the origin of the current coordinate frame to the previous frame, measured along the current joint axis.
- Joint angle (θ) - the angle between two consecutive x-axes, measured about the common normal or current z-axis.

By systematically applying these parameters to each joint in a robot's kinematic chain, it is possible to develop a comprehensive mathematical model that represents the spatial relationship and movements between all parts of the robot.

Overall, the DH parameters provide a robust and consistent way to describe a robot's structure and motion, and they are widely used in both academic research and practical applications of robotics.

Two types of DH parameters exist: the standard DH parameters and the modified DH parameters. In the next subsections, the details of these two types will be explained.

2.7.1 Standard Denavit-Hartenberg Parameters

The standard DH parameters provide an efficient method to describe the geometric relationship between neighbouring links in a robotic manipulator. The system employs four parameters to characterise each link: two parameters capture the link's length and twist (describing displacement), while the other two specify the joint angle and offset (describing rotation). A key characteristic of the standard DH parameters is the placement of reference frames. The frame's origin is aligned with the joint axis, and the z-axis follows the direction of the joint's motion. The x-axis, by contrast, is positioned along the common normal between two consecutive joint axes, effectively defining the link. For the standard DH parameter, the related parameters are depicted below according to Figure 2.2.

1. The link length a_i represent the distance along the x_i axis, from z_{i-1} to z_i.
2. The link twist α_i represent around the x_i axis, the rotation degree from z_{i-1} axis to z_i axis.
3. The link offset d_i, represents the distance along the z_{i-1} axis from x_{i-1} axis to x_i axis.

4. The joint angle, denoted as θ_i, represents the specific rotation around the z_{i-1} axis that causes the x_{i-1} axis to rotate and align with the x_i axis.

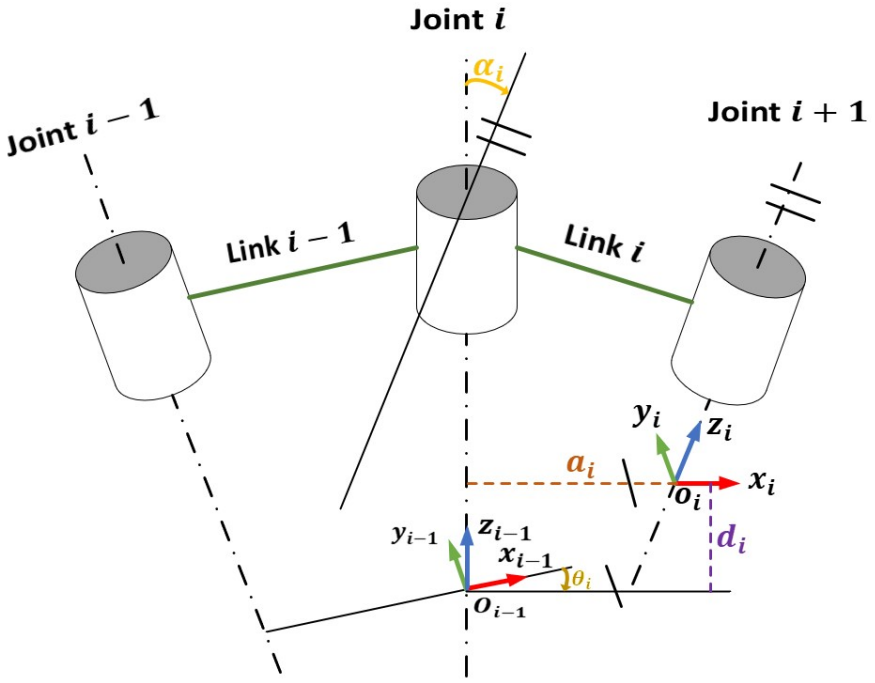


Figure 2.2: Standard DH parameters

To obtain the mathematical relationship between the coordinate system at the end of the robot and the world coordinate system, a coordinate system is established at the end of each link. Then, the DH parameter method is used to determine the mutual relationship between the links. The homogeneous transformation matrix in the standard DH parameter between link coordinate systems can be described as follows:

$$^{i-1}T_i = Rot_{z_{i-1}}(\theta_i) \cdot Trans_{z_{i-1}}(d_i) \cdot Trans_{x_i}(a_i) \cdot Rot_{x_i}(\alpha_i) \tag{2.1}$$

$$\begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.2}$$

where $Rot_{z_{i-1}}(\theta_i)$ represents the rotation by θ_i degrees about the Z_{i-1} axis; $Trans_{z_{i-1}}(d_i)$ denotes the translation of d_i along the z_{i-1} axis from the x_{i-1} axis to the x_i axis. Similarly, $Trans_{x_i}(a_i)$ indicates the translation of a_i along the x_i axis from the z_{i-1} axis to the z_i axis, and $Rot_{x_i}(\alpha_i)$ signifies the rotation by α_i degrees about the x_i axis from the z_{i-1} to the z_i axis.

2.7.2 Modified Denavit-Hartenberg Parameters

The modified DH parameters were developed after the introduction of the standard DH parameters. These variants modify the placement of the x-axis and, consequently, the frame assignment. This adjustment simplifies the description of certain manipulators and can offer computational efficiencies. By allowing the joint and link axes to coincide, the modified DH parameters provide a more intuitive definition of the link and robotic structure than the standard parameters do. For the modified DH parameters, the following parameters should be considered. The related parameters for the standard DH parameter are depicted below, according to Figure 2.3.

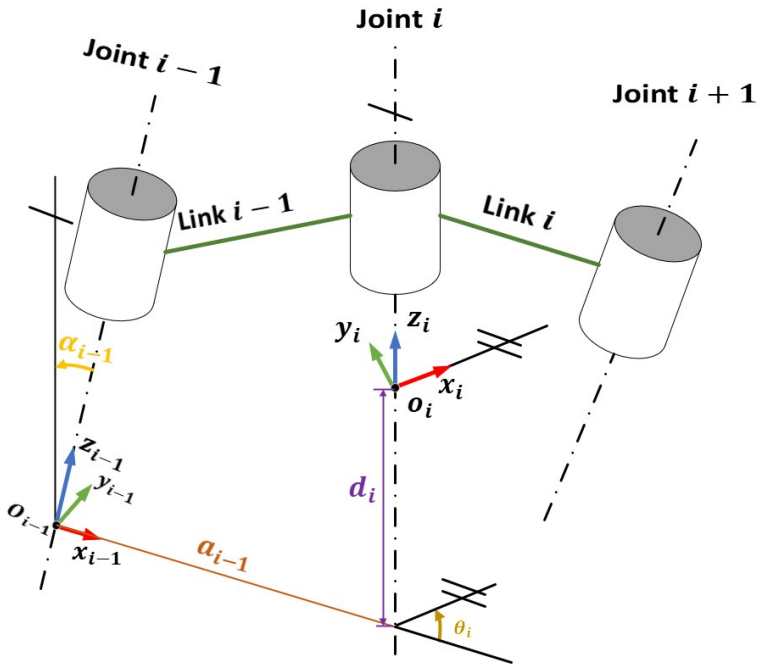


Figure 2.3: Modified DH parameters

1. The link length a_{i-1} represent the distance along the x_{i-1} axis, from z_{i-1} to z_i.

2. The link twist α_{i-1} represent around the x_{i-1} axis, the rotation degree from z_{i-1} axis to z_i axis.

3. The link offset d_i, represents the distance along the z_i axis from x_{i-1} axis to x_i axis.

4. The joint angle, denoted as θ_i, represents the specific rotation around the z_i axis that causes the x_{i-1} axis to rotate and align with the x_i axis.

The transformation matrix between coordinate $i-1$ and i can be described below:

$$^{i-1}T_i = Rot_{x_{i-1}}(\alpha_{i-1}) \cdot Trans_{x_{i-1}}(a_{i-1}) \cdot Rot_{z_{i-1}}(\theta_i) \cdot Trans_{z_{i-1}}(d_i) \tag{2.3}$$

$$^{i-1}T_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \sin\theta_i \cos\alpha_{i-1} & \cos\theta_i \cos\alpha_{i-1} & -\sin\alpha_{i-1} & -\sin\alpha_{i-1} d_i \\ \sin\theta_i \sin\alpha_{i-1} & \cos\theta_i \sin\alpha_{i-1} & \cos\alpha_{i-1} & -\cos\alpha_{i-1} d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

where $Rot_{x_{i-1}}(\alpha_{i-1})$ denotes a rotation of α_{i-1} degrees about the x_{i-1} axis; $Trans_{x_{i-1}}(a_{i-1})$ signifies translation by a_{i-1} along the x_{i-1} axis, moving from z_{i-1} to z_i axis; $Rot_{z_{i-1}}(\theta_i)$ indicates a rotation of θ_i degrees around the z_{i-1} axis; and $Trans_{z_{i-1}}(d_i)$ describes a translation by d_i from the x_{i-1} to the x_i axis.

DH parameters have been widely applied in the industry. Klug et al. [82] detailed a comprehensive assortment of DH parameters for a prototypical robotic total station. Subsequently, Flanders et al. [83] subsequently presented an introduction to virtual reality-based educational instruments. These tools utilise the Virtual Reality Modelling Language and an animation toolbox for the study of forward kinematics, adhering to the Denavit-Hartenberg convention. Additionally, Shen et al. [84] proposed a model crafted using the Denavit-Hartenberg parameter methodology. This model aims to scrutinise both the forward and inverse kinematics of the robot, with the ultimate objective of optimising the robot's trajectory.

2.8 Multi-Objective Optimisation

Multi-objective optimisation, often also termed multi-criteria or multi-attribute optimisation, addresses problems with multiple conflicting objectives. This approach aims to find solutions that strike a balance or trade-off, ensuring that optimising one objective does not negatively impact another. Solutions that achieve this equilibrium are termed Pareto-optimal, a concept named after Vilfredo Pareto, the Italian economist who introduced it [85].

In the context of manufacturing, multi-objective optimisation becomes particularly relevant for tasks such as asset selection and layout optimisation. During asset selection, considerations

such as cost, efficiency, and longevity might conflict, necessitating a balanced approach. Similarly, layout optimisation might require trade-offs between cycle time, energy consumption and safety. A single optimisation objective often falls short of capturing the multifaceted nature of these complex problems. Hence, understanding and applying multi-objective optimisation techniques are pivotal in achieving holistic and effective solutions in these domains.

In a manufacturing scenario, an illustrative example involves balancing the twin objectives of minimising energy consumption and optimising robotic manoeuvrability, both of which are crucial. However, achieving harmony between these goals can be challenging. Certain manoeuvres intended to enhance robotic efficiency might lead to increased energy usage. On the other hand, strategies focused on energy conservation could potentially impede the robot's range of motion or reduce its operational speed.

In this context, a solution is termed Pareto-optimal if no other feasible solution can improve one of the objectives without worsening the other. The Pareto front [86], in this case, would represent the trade-offs between energy efficiency and robot manoeuvrability. Decision-makers equipped with this range of Pareto-optimal solutions can then make informed choices based on the specific needs and constraints of the manufacturing setup.

Decision-making in multi-objective optimisation can be broadly categorised into two primary modes: "a priori" and "a posteriori" [87]. In the "a priori" method, preferences or weights for each objective are established upfront, and based on these predefined weights, the multi-objective problem is converted into a single-objective challenge. The solution derived using this method is expected to align closely with the initial preferences. By contrast, the "a posteriori" method does not require preferences or weights to be set at the outset. It focuses on finding a set of Pareto-optimal solutions without a specific preference structure. Once these solutions are identified, decision-makers can then choose the most suitable one based on their preferences. Scalarisation is one technique used in this approach, where various weight combinations are applied to the objectives to generate different solutions. This method allows for a more flexible and comprehensive exploration of potential solutions, especially when uncertainty arises about the objectives or when a diverse set of optimal solutions is desired.

Multi-objective optimisation finds extensive applications in manufacturing. For instance, in the realm of asset selection, multi-objective optimisation has proven to be invaluable [88, 89]. As asset selection significantly influences manufacturing efficiency, costs, and product quality,

striking a balance among various objectives is essential. These objectives might include minimising costs, reducing energy consumption, and ensuring the long-term sustainability of assets. By harnessing the power of multi-objective optimisation, industries are equipped to make well-informed decisions regarding which assets to invest in, thereby achieving a harmonious balance between cost efficiency, energy conservation, and sustainability.

Layout configuration optimisation, another critical aspect of manufacturing, also extensively employs multi-objective optimisation principles. This area is not solely concerned with the arrangement of machinery and equipment; it is also about navigating challenges such as optimising space and ensuring efficient material flow, minimising movement distances, maximising the utilisation of available space, and ensuring robot manoeuvrability. Through multi-objective optimisation, a holistic solution can be attained that meets these diverse needs [64, 90, 91].

While multi-objective optimisation has shown promise in asset selection and layout optimisation, its application presents unique challenges. The complex nature of defining and prioritising objectives in these domains can lead to oversimplification or misrepresentation of the real-world scenario [92]. The multi-modal nature of the solution space, especially in layout optimisation, can pose difficulties in effectively navigating the Pareto front, potentially leading to suboptimal solutions [93]. Additionally, the computational overhead of evaluating multiple objectives simultaneously, given the vast combination of assets and layouts, can be prohibitive, especially for real-time decision-making [94]. Standard benchmarks and performance metrics tailored for these specific applications are also lacking, making it challenging to assess and compare the effectiveness of different multi-objective optimisation algorithms [95].

In conclusion, methods such as multi-objective and Pareto optimisation offer invaluable frameworks for navigating intricate problems marked by multiple conflicting objectives. By presenting an array of optimal solutions, these techniques empower decision-makers to select the most appropriate solution in line with their distinct requirements and prevailing conditions. The relevance of these techniques is particularly pronounced in sectors such as manufacturing, where they play a pivotal role in refining processes and bolstering efficiency.

2.9 Chapter Summary

Chapter 2 delves into the complexities and challenges of robotic assembly reconfiguration, building on the foundation presented in Chapter 1, which highlights three pivotal research questions. The chapter provides an overview of the existing methodologies related to the research questions, highlighting their limitations. It also presents essential techniques to improve the understanding of the challenges in robotic assembly reconfiguration. The fundamental aspects of these research questions, along with the current strategies and their limitations, are summarised as follows:

1. The RQ1 centres on the efficacious capture and portrayal of knowledge from diverse sources in robotic assembly reconfiguration. Despite advancements in ontologies, semantics, and, notably, knowledge graphs, discernible gaps persist. Many prevailing resource descriptions remain domain-specific, lacking the desired breadth and holistic perspective. A solution that seamlessly amalgamates static and dynamic parameters is also lacking. Knowledge graphs, symbolic of Industry 4.0, hold promise for integrated and exhaustive knowledge management. However, challenges loom, notably due to a lack of standardisation and potential integration complexities.

2. The RQ2 explores the nuanced process of asset selection within robotic assembly cells. The shift from intuition-driven methodologies to predictive analytics is significant. Current strategies, while promising, exhibit limitations: an over-reliance on historical data, limited synergy with broader industrial systems, and potential scalability concerns. Furthermore, multi-objective optimisation emerges as an instrumental paradigm. By balancing conflicting objectives, such as cost and energy consumption, this approach aims to provide a robust framework for asset selection. However, the inherently dynamic nature of these decisions, coupled with the intricacies of setting and harmonising objectives, introduces inherent challenges.

3. The RQ3 delves into the complexities of layout optimisation in robotic assembly systems. In the context of Industry 4.0, there's a growing appreciation for simulation tools due to their ability to offer virtual testing and adjustments without real-world disruptions. A key aspect to consider is the DH parameters, which provide detailed insights into a robot's movement and flexibility. This level of detail is vital for improving the robotic

assembly process. However, current methods have their drawbacks: they often depend heavily on expert knowledge, need large datasets, can face difficulties adapting in real-time, and sometimes falter with problems that have multiple objectives. Additionally, research seems to concentrate on either the individual machine or the larger system, with little integration of the two. Also, while many studies focus on machining processes, there's a noticeable gap in research specifically targeting the layout optimisation of robotic assembly cells.

Chapter 3

Reconfiguration Framework

To address the three research questions (RQ1, RQ2, and RQ3) and fulfil the objectives presented in Chapter 1, and to overcome the limitations of existing approaches detailed in Chapter 2, this PhD thesis proposes a reconfiguration framework for robotic assembly cells and investigate its key perspectives in the frame for the robotic assembly cell including synergising ontology models, knowledge graph and artificial intelligence. This framework mainly comprises three components: the experience databank component, the optimal manufacturing asset selection component, and the layout configuration optimisation component, as illustrated in Figure 3.1.

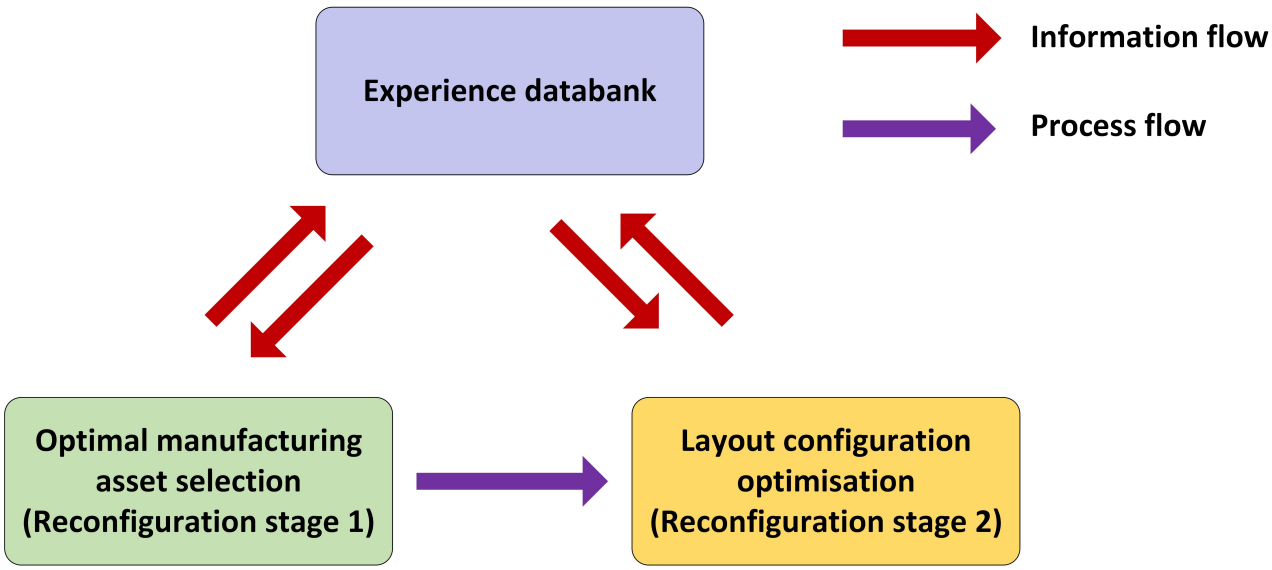


Figure 3.1: Schematic representation of the research's key components

.

3.1 The Proposed Methodology

Historically, manufacturing reconfiguration has been navigated using methodologies that lean heavily on manual insights and human expertise. Recognising the limitations of such an approach, this PhD thesis introduces the experience databank. This innovative tool is not just a repository but a transformative solution, capturing the essence of manual knowledge and offering a streamlined methodology for manufacturing asset selection and layout optimisation. The experience databank operates on two primary axes: manufacturing asset selection and layout optimisation. Based on the above context, the major contributions of this PhD research are four-fold: the methodology of development of the experience databank, the methodology for optimal manufacturing asset selection, the methodology of layout configuration optimisation, and the fourth contribution, which involves the practical validation of the aforementioned methodologies through software implementation and a series of industry use-case examinations.

Examining the interrelation between these components and adapting it in the robotic assembly cell reconfiguration, a more detailed framework is presented in Figure 3.2.

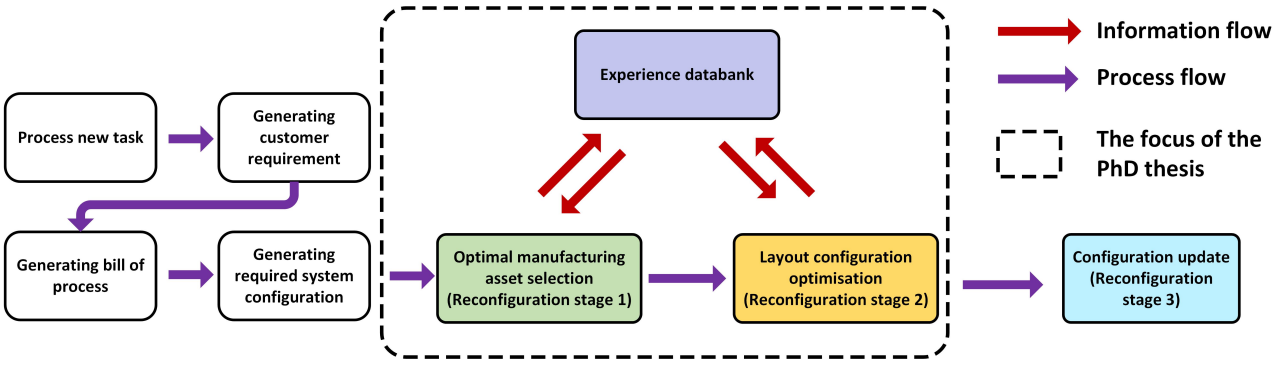


Figure 3.2: The three contributions to robotic assembly cell reconfiguration

This framework outlines how to incorporate the first three major components into the robotic assembly cell's operations to adapt to new process needs. When a new task is introduced, it is processed to generate the customer requirement, bill of process, and required system configuration. Upon reaching the pivotal component of this PhD thesis, the experience databank intervenes, furnishing recommendations for optimal manufacturing asset selection. Once assets are selected, the experience databank provides further guidance for layout configuration optimisation. After the layout is optimised, the real equipment configuration is updated accordingly. The first three principal contributions of this research are methodically explored and detailed in Chapters 4, 5, and 6, respectively. The fourth contribution, which focuses on the real-world

application and validation of these methodologies, is thoroughly discussed in Chapters 7 and 8, underlining its significance in the practical realm. Figure 3.3 details the UML activity diagram of the proposed framework.

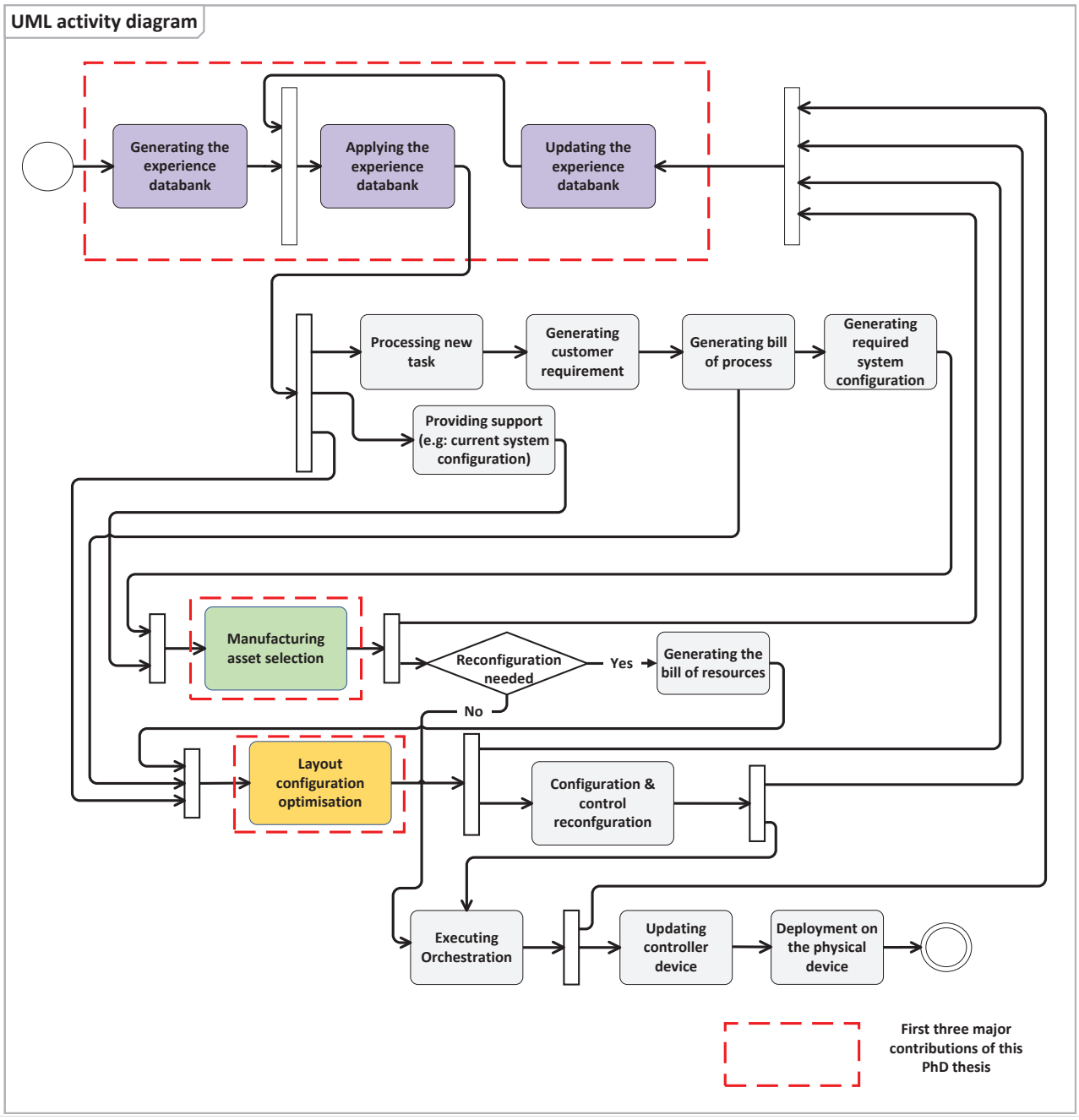


Figure 3.3: UML activity diagram of the reconfiguration framework

As mentioned before, the body of work presented in this thesis is composed of three core methodological contributions, each addressing a distinct challenge in the reconfiguration of robotic assembly systems for advanced manufacturing. To ensure that these contributions are not only theoretically sound but also practically viable, a fourth crucial component has been introduced, which encompasses the software implementation and its validation through use cases.

The four major components (i.e., three methodological contributions and one implementation and validation component) of this PhD thesis are elaborated below:

1. **Building and Updating the Experience Databank (Contribution 1 to Address Objective 1):** The research introduces the concept of an "experience databank" — a dynamic repository that captures details about tasks, processes, assets, capabilities, and the nuances of reconfiguration within robotic assembly cells. This databank, a pivotal element in the infrastructure, supports a continuous evolution and fine-tuning of knowledge. It is developed using an ontology model for a consistent schema and knowledge graph techniques for nuanced data interrelations. The databank is designed to offer:

    - Enhanced data management through efficient algorithms for data capture, storage, and retrieval. This is achieved via a vendor-neutral ontology model, ensuring a uniform approach to data representation and processing. The model's robust structure underpins the effective management and utilisation of data within robotic assembly cells.
    - Improved decision-making capabilities by integrating advanced data management practices. The databank leverages a knowledge graph-centric approach to maintain data reliability, accuracy, and relevance, thereby facilitating informed and timely decisions in the manufacturing process.

2. **Optimal Manufacturing Asset Selection Methodology (Contribution 2 to Address Objective 2):**

    This methodology aims to identify the most appropriate asset to meet new process requirements. Comprising both the recognition and evaluation of potential assets, it supports swift adaptations to changing process requirements in flexible manufacturing systems. Knowledge graphs handle data storage and management, whilst multi-criteria decision-making algorithms evaluate assets. This methodology is broken down into:

    - The initial step involves the creation of selection algorithms accounting for diverse performance metrics and essential elements such as cost, compatibility, and availability. Such algorithms ensure the alignment of selected assets with product requirements.
    - The assessment process benefits from decision-making algorithms, harmonising various objectives and limitations. This comprehensive evaluation of potential assets

draws on their specification, capacity, and reconfiguration model.

3. **Layout Optimisation Methodology (Contribution 3 to Address Objective 3):** A methodology for layout optimisation is devised, focusing on ascertaining the optimal placement of assets. Knowledge graphs manage data storage, while multi-criteria decision-making algorithms identify the optimal layout. The incorporation of simulation environments and modular artificial intelligence during the reconfiguration phase ensures accuracy and flexibility. This methodology comprises:

   - The phase begins with the optimisation of asset positioning within the assembly cell, subsequently updating the configuration settings to reflect the optimised layout. This approach considers varied configurations and layout goals such as energy consumption, robot manoeuvrability, and cycle time.
   - A simulation-based strategy is introduced and designed with a strong emphasis on interoperability. It allows for robust evaluations of layout configurations at both the machine and system levels prior to their real-world deployment. This interoperable framework ensures the seamless integration of the simulation environment with various algorithms, bolstering effective communication and interaction with external systems. Different simulation software tools have been incorporated to guarantee scalability. The approach not only promises a holistic insight into assembly environments but also aids in the selection of the most efficient layouts.

4. **Software Implementation and Use Case Validation (Contribution 4 to Address Objective 4):**
   To validate the proposed methodology encompassing the first three major contributions, a meticulous software development process is executed, resulting in a bespoke suite that integrates functionalities for the experience databank, modular manufacturing asset selection, and layout optimisation. This software suite is subjected to rigorous scrutiny through three distinct use cases, each designed to challenge and assess the robustness and applicability of the methodologies within real-world manufacturing settings.

3.2 Validation Methods

Within this chapter, the validation criteria are delineated, serving to ascertain the efficacy of the three principal methodological contributions of this doctoral thesis. Each contribution, alongside its ancillary components, is meticulously tethered to specific validation criteria, ensuring the attainment of the designated objectives. An in-depth exposition of these criteria and their corresponding use case validations are provided in Chapter 8, wherein three carefully selected use cases are expounded. Each case has been strategically chosen to satisfy distinct criteria, thereby affirmatively meeting the collective objectives of the research.

It should be highlighted that the fourth contribution of this thesis, encompassing software development and use case validation, does not require independent validation against the established criteria. This component is pivotal in demonstrating the practical application of the first three contributions. It functions as the medium through which the developed methodologies are applied and validated against real-world industrial scenarios, inherently substantiating the first three objectives rather than being subjected to validation itself.

3.2.1 Validation of Methodology for Building and Updating Experience Databank

Contribution 1 focuses on developing a vendor-neutral ontology model that effectively describes manufacturing capabilities, capacities, and reconfiguration strategies. The databank leverages a knowledge graph-centric approach to maintain data reliability, accuracy, and relevance, thereby facilitating informed and timely decisions in the manufacturing process. The validation measures to demonstrate that this contribution fulfils Objective 1 are as follows:

- **Criterion 1.1: Vendor Neutrality**

  The vendor neutrality of the ontology model will be validated by building the model without considering vendor-specific information. It will also be assessed by applying the model to a diverse set of industry use cases involving multiple vendors. The methodology to satisfy this criterion is detailed in Section 4.2, and the use cases for validating the methodology are comprehensively undertaken in Sections 8.1, 8.2, 8.3.

- **Criterion 1.2: Information Modelling**

  This can be validated by the successful modelling of the robotic assembly reconfiguration-related information. The methodology to satisfy this criterion is detailed in Section 4.2, and the use cases for validating the methodology are comprehensively undertaken in Sections 8.1, 8.2, and 8.3.

- **Criterion 1.3: Handling of Data from Diverse Sources**

  The capability of the methodology to handle data from diverse sources will be confirmed by showcasing its operation across different data types. The approach for achieving this is elaborated in Section 4.2, and the use cases for validating this methodology are comprehensively undertaken in Sections 8.1, 8.2, and 8.3.

- **Criterion 1.4: Reasoning Based on the Ontology**

  This will involve the test of checking if the ontology reasoning works and if the implicit information can be found via ontology reasoning. The methodology to satisfy this criterion is detailed in Section 4.2. The use cases for validating this methodology are comprehensively undertaken across multiple sections, namely Section 8.1, Section 8.2, and Section 8.3.

3.2.2 Validation of Optimal Manufacturing Asset Selection Methodology

The second contribution is centred on developing an effective manufacturing asset selection and evaluation methodology. The validation criteria for this contribution are as follows:

- **Criterion 2.1: Adapting to New Process Requirement**

  This criterion can be validated by providing the different process and product requirements for the proposed framework. Then, the framework enables the adaptation to the changing requirements. The methodology to satisfy this criterion is detailed in Section 4.2. The use cases of validating this methodology are comprehensively undertaken across multiple sections, namely Section 8.1, and Section 8.3.

- **Criterion 2.2: Capability Assessment**

  This criterion evaluates assets' suitability for specific process requirements. The method-

ology focuses on a thorough assessment of asset attributes and performance to ensure that selected assets align with operational needs. Details of this assessment are provided in Sections 5.2 and Section 5.3. The use cases of validating this methodology are comprehensively undertaken across multiple sections, namely Section 8.1 and Section 8.3.

- **Criterion 2.3: Modular Manufacturing Asset Selection**

  The modularity of the manufacturing asset selection methodology can be validated by ensuring that the methodology can be easily integrated into various algorithms. This will involve testing the methodology with different algorithms and assessing the ease and effectiveness of the integration. The methodology to satisfy this criterion is detailed in Section 5.4, and the use case for validating this methodology is presented in Section 8.3.

- **Criterion 2.4: Multi-Criteria Manufacturing Asset Selection**

  The asset prioritisation process employs an multi-criteria decision-making (MCDM) approach to systematically rank the identified candidate assets based on multiple performance criteria. The proposed methodology must be able to support MCDM. The methodology to satisfy this criterion is detailed in Section 5.4. The use case for validating this methodology is presented in Section 8.3.

- **Criterion 2.5: Synergies with the Systems**

  This criterion aims to validate the synergies with other assets in manufacturing asset selection. In manufacturing asset selection, the single machine and the synergies with other machines should be considered. The methodology to satisfy this criterion is detailed in Section 5.4. The use case for validating this methodology is comprehensively examined in Section 8.3.

3.2.3 Validation of Methodology for Layout Optimisation

The third contribution targets the creation of an optimisation methodology for layout configuration. The validation criteria for this contribution are as follows:

- **Criterion 3.1: Multi-Criteria Layout Optimisation**

  This criterion will be validated by providing more than one optimisation objective in the layout optimisation. The methodology related to this criterion is elucidated in 6.2, and the use case for validating the methodology is presented in Section 8.2.

- **Criterion 3.2: Modular Layout Optimisation**

  The modularity of the proposed framework can be validated by verifying the ease of integration and replacement of different optimisation algorithms. This will involve testing the framework with different algorithms and evaluating its ability to adapt to changes in the algorithms. The methodology related to this criterion is elucidated in Section 6.2, and the use case for validating the methodology is presented in Section 8.2.

- **Criterion 3.3: Interoperability**

  The successful interoperability of the simulation environment with the algorithm and its ability to communicate effectively with the external world will be validated, ensuring that they can effortlessly interact. This will entail a series of tests in which the algorithm is executed within the simulation environment, assessing its capability to communicate with the environment and utilise its features effectively. The methodology related to this criterion is elucidated in Section 6.2, and the use cases for validating the methodology are comprehensively undertaken in Section 8.1 and Section 8.2.

- **Criterion 3.4: Scalability**

  This criterion aims to validate the generalisation of the layout configuration framework. Validation can be achieved by applying the proposed layout reconfiguration framework to different layout optimisation problems, not only at the machine level but also at the system level. The methodology related to this criterion is elucidated in Section 6.2. The use cases for validating this methodology are comprehensively undertaken across multiple sections, namely Section 8.1 and Section 8.2.

3.3 Chapter Summary

This chapter introduced a framework for reconfiguring robotic assembly processes. This framework is structured around three core components:

1. **Methodology to Build and Update the Experience Databank:** This component introduces the concept of a vendor-neutral ontology model and its synergy with the knowledge graph.

2. **Methodology for Optimal Manufacturing Asset Selection:** This component outlines the process of identifying and evaluating candidate assets.

3. **Methodology for Layout Configuration Optimisation:** This component focuses on multi-objective and simulation-based optimisation.

The aforementioned three components form the foundational triad of contributions of this PhD thesis, while the fourth pivotal contribution is the software development process accompanied by use cases specifically devised to validate the methodologies established by the initial three contributions.

The ontology model, detailed in Chapter 4, is posited as an instrumental tool in robotic reconfiguration. It is the key element of building the experience databank with a knowledge graph. The experience databank is intended to streamline decision-making complexities in manufacturing asset selection, evaluation, and layout optimisation.

The manufacturing asset selection methodology, discussed in Chapter 5, considers various criteria such as cost, cycle time and energy consumption. Its aim is to identify an efficient combination of assets to meet process requirements.

The methodology for layout configuration optimisation is tailored to user-defined Key Performance Indicators (KPIs) and allows for algorithmic flexibility based on specific use cases. This will be elaborated upon in Chapter 6.

Representative scenarios, such as robotic manufacturing and industrial production cells, will serve as the backdrop for the validation process. These scenarios aim to embody the unique challenges and nuances of reconfiguring robotic assembly processes. Chapter 7 and Chapter 8 describe the software development process and use cases that are fundamental to the three major contributions of this research.

In conclusion, this chapter has provided an overview of a framework developed for reconfiguring the robotic assembly process. This framework is intended to enhance both manufacturing asset selection and layout configuration processes by leveraging an experience databank.

Chapter 4

Ontology Model and Experience Databank

This chapter is a crucial thesis component because it addresses the RQ1. It introduces an ontology model and explains how to build the experience databank through the knowledge graph with the help of the ontology model, thus achieving Objective 1.

4.1 Introduction

The relationship between experience databank, knowledge graph and ontology model is depicted in Figure 4.1. As mentioned in Chapter 2, the knowledge graph consists of the schema and entity layers. The ontology model is necessary to build the schema layer of the knowledge graph. This chapter proposes a unified formal ontology model integrating capacity, capability, and reconfiguration information. Employing this model to represent manufacturing information can help make more efficient, accurate, and timely decisions. Furthermore, an approach to building the entity layer of the knowledge graph has also been proposed. After the entity and schema layers of the knowledge graph are built, the experience databank is achieved based on the generated knowledge graph.

To ensure that the ontology model effectively fulfils the needs of the robotic assembly reconfiguration domain, it must:

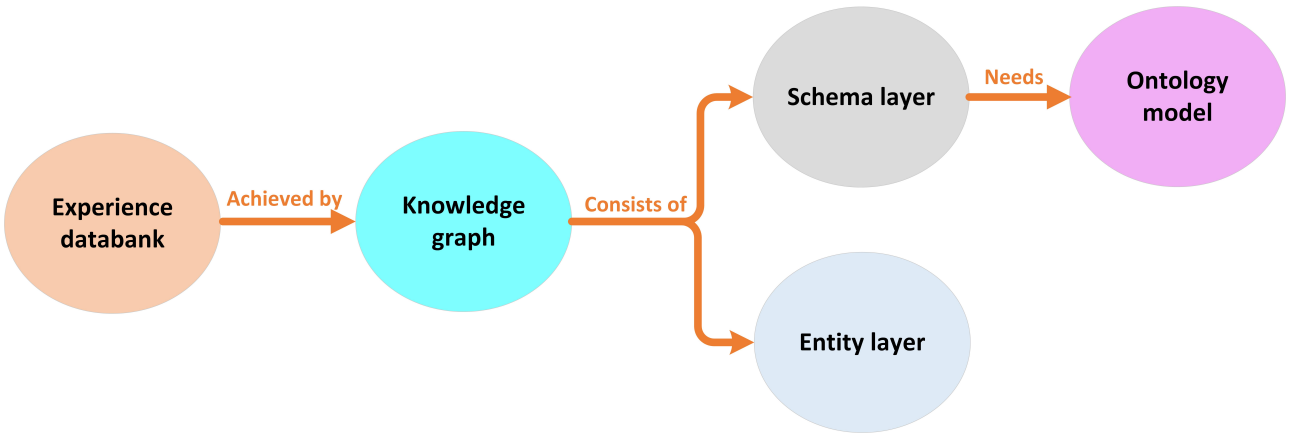


Figure 4.1: Relationship between experience databank, knowledge graph and ontology model

- Be vendor-neutral.
- Accurately represent the capability, capacity, and reconfiguration of the robotic assembly cell.
- Enable the processing of heterogeneous data through a knowledge graph-based approach
- Facilitate reasoning based on the ontology model.

In the subsequent chapters, the ontology model's design and the ontology model's application with a knowledge graph-based approach to building the experience databank are evaluated against the above-mentioned requirements to ensure the model's efficacy and relevance in the domain.

4.2 Development of Ontology Model and Experience Databank in Knowledge Graph

In this section, the process of how to build the experience databank is proposed. At first, the knowledge graph-building approach is introduced and chosen to build the knowledge graph for the robotic assembly reconfiguration. Second, the information required for the ontology model to construct the knowledge graph is detailed. Additionally, the relationships among the various pieces of information within the model are described. Then, as the thesis focuses on the reconfiguration of robots, the robot information and reconfiguration models are described. Finally, the application of the generated knowledge graph (experience databank) is described.

4.2.1 Knowledge Graph Building Approach

The creation of a knowledge graph, as identified in the literature, involves articulating both a schema and an entity layer. A strategic approach is required to develop these layers when the knowledge graph is tailored towards reconfiguration tasks.

Two primary methodologies exist for constructing knowledge graphs: a top-down and a bottom-up approach. In the top-down process, domain experts and existing datasets are employed to define the schema layer a priori, which then guides the population of the entity layer [96]. This method is prevalent for designing knowledge graphs that cater to specific domains or applications and relies heavily on expert input. On the other hand, the bottom-up approach initiates the extraction of entities and their relationships from a myriad of data sources, both structured and unstructured. This is followed by the gradual formulation of the schema layer, informed by the insights garnered from the entity information [97].

Adopting solely the top-down or bottom-up approach comes with its own set of challenges. The top-down method, while ensuring the precision of data, often requires significant time and financial investment due to its reliance on domain expertise. In contrast, the bottom-up method is less resource-intensive but hinges on the availability of extensive data, which can pose significant obstacles, particularly within the manufacturing sector. The approach recommended in this study aims to synergise the decision-making process and enhance reconfiguration efficacy by integrating both methods. This hybrid strategy mitigates the limitations inherent in using each approach in isolation, as elaborated in [98]. Sections 4.2.2 and 4.2.3 detail the methodologies to build the schema layer and the entity layer of the knowledge graph, respectively. Section 4.2.4 describes the application of the generated knowledge graph (experience databank).

4.2.2 Ontology Model as the Schema Layer of the Knowledge Graph

An ontology model named the "Ontology model of capability, capacity, and reconfiguration (OCCR)" is developed to construct the schema layer of the knowledge graph. It is based on Järvenpää's model [14] but incorporates additional semantic models and refines existing ones; for example, it incorporates a more detailed capability model, including metrology capability, and information regarding reconfiguration, capacity, and tasks. Within the purview of

the OCCR model is the formal modelling of the capabilities, capacities, and reconfiguration information of robotic assembly reconfiguration, along with their associated models. This is achieved through automated analysis of the utilisation of available assets and the autonomous allocation of capacity to optimise that utilisation in response to fluctuating market demands. The OCCR model comprises seven semantic models: task, product, process, capability, capacity, assets, and reconfiguration models, each comprising the relevant ontology classes. The data is stored as triples in the knowledge graph. A symbolic representation of these semantic models is presented in Table 4.1, with a more in-depth description of each model provided in the subsequent sub-sections. The interrelationships among these semantic models are visually represented in Figure 4.2.

Table 4.1: Seven semantic models and their symbolic representation

Models	Symbolic Representation
Task	TAS
Product	PRT
Process	PRS
Capability	CAB
Capacity	CAP
Assets	ASS
Reconfiguration	REC

4.2.2.1 Task Model

The *task model* describes the information about how customer orders and requests that are initiated internally, within the factory, are processed. It represents the tasks or activities necessary to complete a customer order or fulfil a request, providing a high-level overview of the steps involved in the production process. The task model is divided into two sub-models: non-reconfiguration-related tasks and reconfiguration-related tasks.

1. **Non-Reconfiguration-Related Task Model (NRT)**

   The NRT defines the non-reconfiguration-related task, which typically originates from the customer and is intended to meet specific requirements for producing the product, such as quantity and timeline. This task can also be interpreted as a production task.

2. **Reconfiguration-Related Task Model (RT)**

   The RT defines the reconfiguration-related task, typically as an internal factory task. This model provides information about reconfiguration, which can take different forms, such as layout reconfiguration, resource selection, and job scheduling.

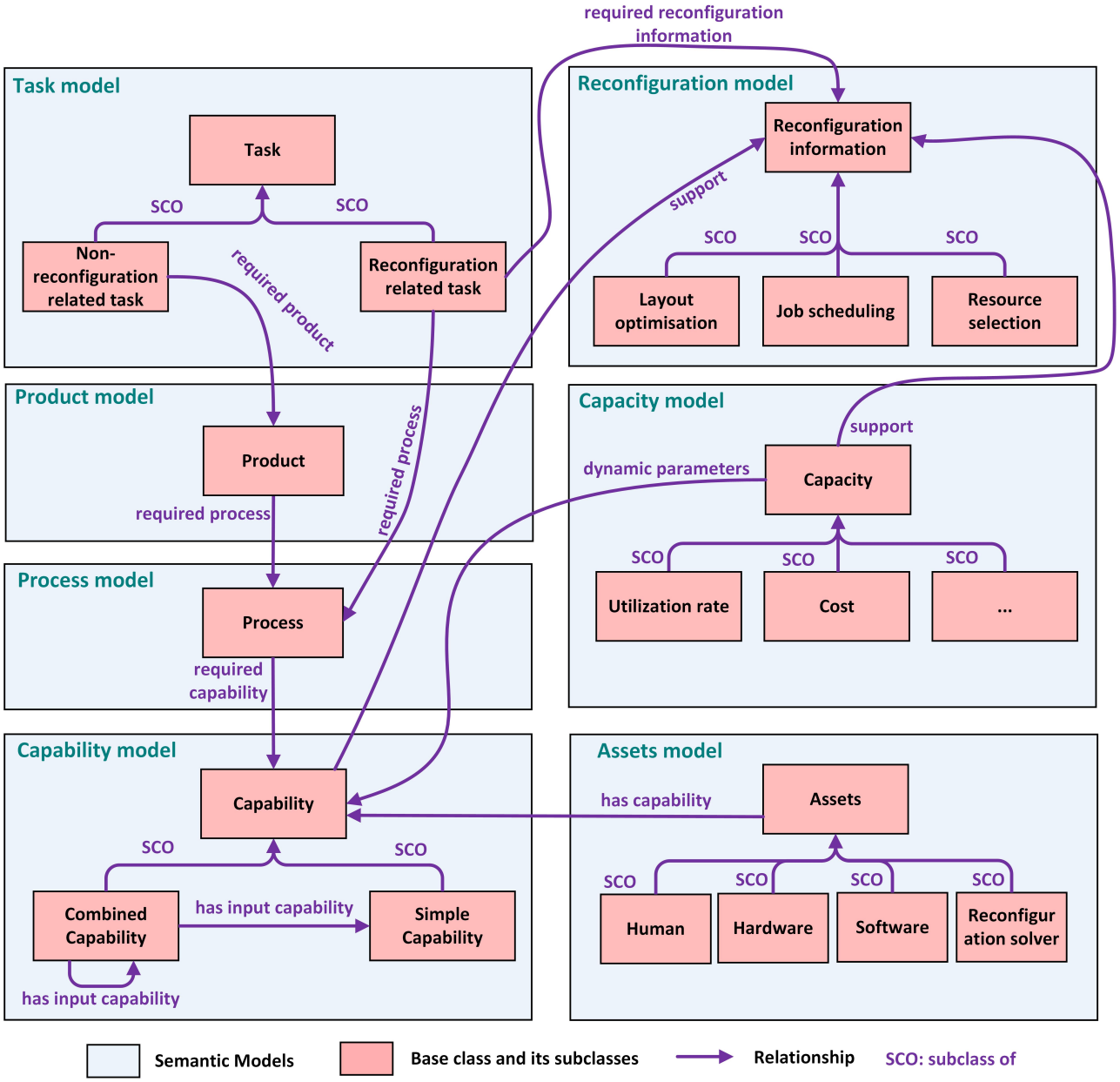


Figure 4.2: Linking semantic models in the optimisation and the decision-making process for the reconfiguration

4.2.2.2 Product Model

The *product model* plays a pivotal role in robotic assembly cells; it represents information about the product in the robotic assembly line. The selection of the most efficient production procedures is heavily influenced by the geometric attributes of the product, such as its size and shape. For instance, if, on the one hand, a product is large or has a complex shape, then robotic manufacturing cells equipped with industrial robots that have a large workspace and high flexibility might be suitable. These robots can manipulate large objects or accurately navigate around complex shapes, making them ideal for these types of tasks. On the other hand, if a

product has intricate geometric features or requires precise assembly, then robotic cells with robots that possess high precision and advanced control features might be a better choice. These robots can handle delicate assembly tasks and accurately follow complex trajectories, thereby ensuring the quality of the finished product. In addition, the product's requirements for assembly and handling can also influence the choice of robotic cells. For example, if the product requires specific positioning or orientation during assembly, then robots with advanced vision systems or force-sensing capabilities might be necessary.

4.2.2.3 Process Model

The *process model* provides a structured representation of the necessary operations and requirements to complete a manufacturing or reconfiguration task step. It explains the various steps involved in a manufacturing task or reconfiguration process, defining the inputs, outputs, and dependencies of each step.

4.2.2.4 Capability Model

A *capability model* in the manufacturing domain represents the capabilities and constraints of a factory in terms of its manufacturing processes and technologies, the available resources and skills, and the regulations and standards with which it must comply. This model provides a comprehensive view of the factory's capabilities and aids in making informed decisions about the products and processes that can be manufactured within the factory. It can also help to identify areas for improvement, such as the acquisition of new resources or technologies or the development of new skills, to enhance the factory's capabilities and competitiveness. It consists of two subclasses: simple capability and combined capability. In the capability model, simple and combined capabilities are linked by "hasInputCapability" relations.

1. **Simple Capability (SC)**

   A simple capability is the capability of a single asset. For example, a fixture has the single capability of "fixturing".

2. **Combined Capability (CC)**

   Combined capabilities are combinations of two or more (simple or combined) capabilities.

They could be divided by functional decomposition into simple, lower-level capabilities. For example, a robot with a finger gripper has the pick-and-place capability, composed of the "force applying" and "moving" capabilities from the robot and the "grasping" and "releasing" capabilities from the finger gripper.

4.2.2.5 Capacity Model

The *capacity model* provides a structured framework that outlines the various KPIs that are used to understand a manufacturing system's capacity constraints and potential. Rather than portraying the performance of the shop floor, the model focuses on illustrating the production capacity of a factory or production line. It presents a view of the available resources (e.g. machines and labour), ongoing production processes, and any existing constraints. This model merely reflects the current manufacturing scenario. Based on this model, algorithms or functions can be developed to analyse further and determine optimal production strategies.

4.2.2.6 Assets Model

In the manufacturing domain, assets encompass a broad range of elements vital for production, from tangible physical objects such as machinery to intangible elements such as software solutions. The *assets model*, therefore, represents these diverse elements to facilitate optimal management and utilisation by organisations. The model classifies assets into four distinct categories:

- **Hardware:** The tangible equipment and tooling used in production processes.
- **Software:** Digital solutions and tools, excluding specialised reconfiguration solvers.
- **Human Workforce:** Human resources and their associated skills, training, and experience.
- **Reconfiguration Solver:** A distinct category given its critical role in the manufacturing setup. While it technically falls under software, its importance and specialised nature merit a separate classification.

Different asset categories represent distinct attributes. For instance, hardware might have maintenance history and location details, while the human workforce could have attributes related to training, skills, and experience. The assets model provides relevant details for each category, thus ensuring comprehensive insights for informed decision-making.

4.2.2.7 Reconfiguration Model

The *reconfiguration model* outlines the attributes that allow an RMS to adjust to changes in customer requests. These attributes include decision variables, optimisation variables, and constraints in reconfiguration scenarios. In the current OCCR model, three forms of reconfiguration are defined: layout optimisation, resource selection for reconfiguration, and job scheduling.

4.2.2.8 Relationships between the Seven Semantic Models

The representation in Figure 4.2 provides guidance on the storage of information as triples in the knowledge graph according to the proposed schema structure. For instance, the relationship between capability and assets is captured using the triple *(Asset, hasCapability, Capability)*. Furthermore, Figure 4.3 displays information related to the subclass of layout optimisation, job scheduling, and resource selection in the reconfiguration model, exemplified by the triple *(Objectives, SCO, Layout Optimisation)*.

These semantic models work together to achieve the decision-making process and the cost-effectiveness of the reconfiguration of the manufacturing system. The reconfiguration- and non-reconfiguration-related tasks have different processes for utilising the semantic models.

When a new task for the manufacturing system arrives, it will be identified by the task model to determine whether it is a non-reconfiguration- or reconfiguration-related task. The steps for utilising the semantic model for enhancing decision-making and the reconfiguration process are different.

For the non-reconfiguration related task, the task model displays the necessary product information and connects it with the product model through the relationship “required product”. The NRT displays details such as product type, quantity, and delivery timeline. Meanwhile, the product model showcases the product’s features, which are linked to the process model

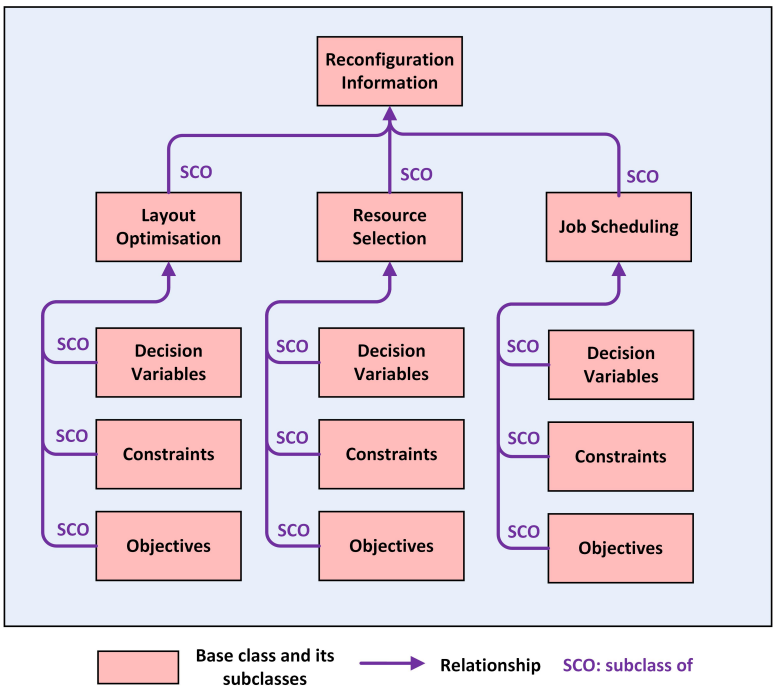


Figure 4.3: Structure of the reconfiguration semantic model

through the relationship "required process". For instance, if the product requires a hole, then the drilling process must be executed to fulfil this requirement.

In the process model, the relevant production process is selected since the task is not reconfigurable. The process model also specifies the varying requirements for each process. Once the process for producing the product is determined, the capability-matching process begins, where the required capabilities to execute the process are identified using information from the process model, such as the accuracy and force required for the process "Inserting".

The asset semantic model links with the capability model, such that once the required capabilities are clear, the candidate assets that meet the specifications can be identified. Additionally, dynamic parameters from the capacity model, such as utilisation rate, cost, and working status, can be considered to optimise the capability-matching process. Finally, the most suitable assets, which exhibit the required capabilities and specifications, are selected.

In the context of reconfiguration-related tasks, the task semantic model serves as a repository for the type of reconfiguration involved. The OCCR model defines three types of reconfiguration, which can be either a single type or a combination thereof. The task model is linked directly to the reconfiguration model, enabling the retrieval of information necessary for reconfiguration based on the type indicated in the task model. The reconfiguration model provides

information about the decision variables, optimisation objectives, and constraints that should be considered during the reconfiguration process. The task model is also linked to the process semantic model through the relationship "required process".

For the RT, two types of processes are required: (1) the reconfiguration solution, such as the "layout reconfiguration process", "resource selection process", and "job scheduling process", and (2) the current process that needs reconfiguration. For instance, as depicted in Figure 4.4, consider a work cell consisting of a robot, a profile board storage rack, profile boards, and a frame on an automated guided vehicle. The robot picks up the profile board from the storage rack and places it on the frame. If the customer requires the layout of the current work cell to be optimised, then the semantic model identifies two types of processes. In this case, the reconfiguration solution process is the "layout reconfiguration process", and the current processes that will be subject to the reconfiguration are "pick profile board" and "place profile board".

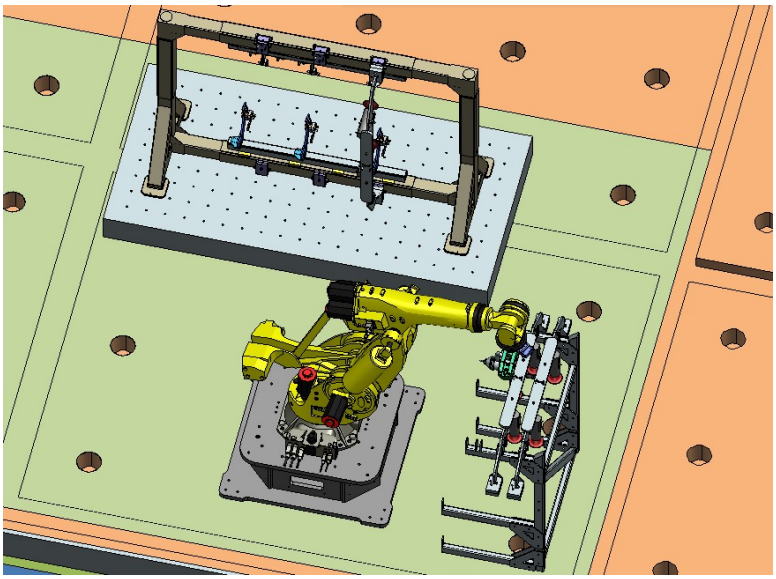


Figure 4.4: Example to show the relationship between the two types of the process in the reconfiguration-related task

Following the clear identification of the two types of processes, the capability-matching process commences. If resource selection is one of the reconfiguration types in the reconfiguration-related task, then the capability-matching process is employed to determine the feasibility and viability of alternative assets replacing existing assets in the production line. For the capability matching of the reconfiguration solution, potential assets that can execute the reconfiguration are sought.

4.2.2.9 Detailed Information of the Reconfiguration Model

The reconfiguration model describes the features that make an RMS dynamic, with the capacity and functionality to adapt to changes in customer requests. The features can be decision variables, optimisation variables, and constraints in the reconfiguration scenarios. In the current OCCR model, three forms of reconfiguration are defined: layout optimisation, resource selection for reconfiguration, and job scheduling. As previously mentioned in Section 4.2.2, a new semantic model has been defined to capture the necessary information for supporting reconfiguration in manufacturing systems. As described in Chapter 2, various types of reconfiguration exist, represented by the subclasses of layout reconfiguration, resource selection, and job scheduling in the current model. These subclasses are included in the reconfiguration semantic model, as shown in Figure 4.2. Decision variables, optimisation objectives and constraints are used as the subclasses of each of the three aforementioned subclasses to describe the reconfiguration information as shown in Figure 4.3.

In manufacturing reconfiguration optimisation, *decision variables* are values that can be chosen or adjusted to optimise the objective, such as cost, efficiency, or production rate. *Optimisation objectives* are the objectives that can be optimised depending on the specific application and goals of the reconfiguration optimisation problem in the manufacturing domain. *Constraints* in the manufacturing reconfiguration domain refer to limitations or restrictions on the decision variables that must be adhered to in order to ensure a feasible and practical solution. A subclass in the asset semantic model is also considered: *Reconfiguration solver*. This subclass represents the enabling technology for achieving the reconfiguration, such as the algorithm, the simulation platform, virtual reality (VR), and augmented reality (AR). These three reconfiguration types in the reconfiguration semantic model are explained next.

4.2.2.9.1 Layout Optimisation In the field of RMSs, the performance of operations on products (or product families) according to their operational requirements is crucial. Layout design and optimisation play a key role in RMSs since these systems require different layout configurations when switching from one product family to another. The following information has been identified as essential and is classified as a subclass in the proposed semantic model:

- **Decision Variables:** Information regarding the pose of assets (e.g. coordinates and

rotation angles) is considered a subclass of the decision variables in optimising manufacturing reconfiguration. This information is classified into coordinates (such as Cartesian coordinates, cylindrical coordinates, polar coordinates, and spherical coordinates) and orientation (such as Euler angles and rotation matrices).

- **Optimisation Objectives:** The optimisation objectives in the semantic model include cost, quality, cycle time, space utilisation, and robot manoeuvrability.

- **Constraints:** Constraints include inequality constraints and equality constraints. They are mathematical relationships used in optimisation problems to ensure that the solutions satisfy certain requirements or limitations. In the context of layout optimisation in manufacturing, these constraints can represent various physical or operational restrictions:

    1. Inequality Constraints: These constraints establish upper or lower bounds on the values of the decision variables or on functions of the decision variables. In the layout optimisation problem, inequality constraints may include the following:

        (a) Non-Collision Constraint (NC): NC ensures that machines do not overlap or collude with each other in the layout. These constraints can be formulated by setting a minimum distance between the edges of any pair of machines.

        (b) Reachability Constraint (RC): RC ensures that machines or workstations are accessible to workers or material handling equipment, such as robots or conveyor systems. These constraints can be formulated by setting a maximum distance between machines or specifying a minimum clearance for pathways or aisles.

        (c) Area Constraint (ARC): ARC ensures that the total area occupied by the machines does not exceed a predefined maximum area. These constraints can be formulated as the sum of the areas of individual machines being less than or equal to the maximum allowed area.

    2. Equality Constraints: These are constraints that require an exact relationship between the decision variables or functions of the decision variables. In the layout optimisation problem, some of the equality constraints may include:

        (a) Resource Allocation Constraint (RAC): This constraint ensures that the total number of certain types of machines or resources within the entire layout is fixed. For example, if a limited number of robotic assembly cells is available,

the layout optimisation problem should include a constraint that ensures the exact number of robotic assembly cells is used in the layout.

(b) Grouping Constraint (GC): This constraint ensures that a specified number of certain types of machines or resources are grouped together within a specific section of the layout. For instance, if a certain manufacturing process requires three specific machines to be located close together for efficiency, the layout optimisation problem should include a constraint that ensures these three machines are grouped together in the layout exactly.

Both inequality and equality constraints help to model the physical and operational limitations of the manufacturing environment and ensure that the resulting layout is practical, feasible, and efficient.

One example of optimising the layout of a manufacturing system is depicted below. The optimisation objectives are:

1. Minimise Space Utilisation (SU)
2. Minimise Cycle Time (CT)
3. Minimise Total Distance between Machines (TD)

The decision variables in this problem are the coordinates and orientation of each machine in the manufacturing system. The optimisation problem can be formulated as:

$$\begin{aligned} \text{Minimise} \quad & f(x) = \{SU(x), CT(x), TD(x)\} \\ \text{subject to} \quad & NC_i(x) \leq 0, \quad i = 1, \ldots, m_1 \\ & RC_j(x) \leq 0, \quad j = 1, \ldots, m_2 \\ & ARC_k(x) \leq A_{\max}, \quad k = 1, \ldots, n_1 \\ & RAC_l(x) = R_l, \quad l = 1, \ldots, n_2 \\ & GC_m(x) = G_m, \quad m = 1, \ldots, n_3 \end{aligned} \tag{4.1}$$

where:

- x: A vector representing the pose of each machine in the manufacturing system, including both coordinates and orientation.

- $f(x)$: A vector containing the objective functions to be minimised, which includes minimising space utilisation, cycle time, and the total distance between machines.
- $NC_i(x)$: The i-th non-collision constraint function that must be satisfied by the pose x.
- $RC_j(x)$: The j-th reachability constraint function that must be satisfied by the pose x.
- $ARC_k(x)$: The k-th area constraint function that must be satisfied by the pose x, with $A_{\max}$ being the maximal allowed area.
- $RAC_l(x)$: The l-th resource allocation constraint function that must be satisfied by the pose x, with R_l being the exact number of a certain type of resource required.
- $GC_m(x)$: The m-th grouping constraint function that must be satisfied by the pose x, with G_m being the exact number of a certain type of resource required in a specific group.
- m_1: The number of non-collision constraints.
- m_2: The number of reachability constraints.
- n_1: The number of area constraints.
- n_2: The number of resource allocation constraints.
- n_3: The number of grouping constraints.

By addressing this optimisation problem, the process of determining the optimal layout for the manufacturing system becomes dynamic and adaptable. It involves updating the decision variables – specifically, the coordinates and rotation angles of the assets within the manufacturing system. By adjusting these variables, a new layout configuration emerges, which is then assessed based on updated optimisation objectives. This approach allows for the efficient reconfiguration of the system's layout to suit different product families or operational requirements. Essentially, each change in the decision variables leads to a potential new layout. The optimality of this layout is evaluated against the current objectives, ensuring that the system remains aligned with changing production goals and constraints. This iterative process of adjustment and evaluation facilitates the continual adaptation of the manufacturing system to meet evolving demands.

4.2.2.9.2 Resource Selection in Reconfiguration Resource selection in reconfiguration refers to the process of choosing the right assets (e.g. hardware, software, or personnel) to implement changes in a system or to apply to processes to achieve a desired outcome. This involves evaluating various options based on factors such as cost, compatibility, performance, and availability and selecting those that best fulfil the needs of the reconfiguration effort. Resource selection aims to ensure that the reconfiguration is carried out efficiently and effectively, with minimal disruption to existing operations. It aims to determine whether the current assets in the production line meet the requirements and if they need to be replaced. The following aspects are essential in this chapter, and thus, this information is stored as the ontology model classes in the model:

- **Decision Variables:** The resource information, number of product types, product requirements, and job information are considered the decision variables in this model. These factors are implemented as subclasses of the decision variables in the proposed semantic model.
- **Optimisation Objectives:** Resource utilisation, cost (investment cost and capital cost), workload, running status of the machines, energy consumption, and remaining useful life are the optimisation objectives.
- **Constraints:** In the context of resource selection, constraints can be categorised into equality and inequality constraints:

  1. Inequality Constraints
     (a) Demand Constraint (DC): DC ensures that the selected resources are sufficient to meet the demand of the reconfiguration task without exceeding the available resources.
     (b) Investment Constraint (IC): IC ensures that the total investment for the selected resources does not exceed the budget allocated for the reconfiguration task.
     (c) Space Constraint (SC): SC ensures that the selected resources can be accommodated within the available space in the production line or facility.
  2. Equality Constraints
     (a) Total Resource Allocation (TRA): TRA ensures that the total number of required resources equals the available resources in the system.

(b) Specific Resource Requirements (SRR): SRR ensures that the selected resources meet the exact specifications or requirements for the reconfiguration task.

As one example, the goal is to optimise resource selection for reconfiguration, considering the following objectives:

1. Minimise Resource utilisation (RU)
2. Minimise Cost (C)
3. Minimise Time (T)

The decision variables in this problem are the resources chosen for the reconfiguration effort. The optimisation problem can be formulated as:

$$\begin{aligned} \text{Minimise} \quad & f(x) = \{RU(x), C(x), T(x)\} \\ \text{subject to} \quad & DC_i(x) \leq 0, \quad i = 1, \ldots, m_1 \\ & IC_j(x) \leq 0, \quad j = 1, \ldots, m_2 \\ & SC_k(x) \leq 0, \quad k = 1, \ldots, m_3 \\ & TRA_l(x) = 0, \quad l = 1, \ldots, n_1 \\ & SRR_m(x) = 0, \quad m = 1, \ldots, n_2 \end{aligned} \tag{4.2}$$

where:

- x: A vector representing the resources chosen for the reconfiguration effort.
- $f(x)$: A vector containing the objective functions to be minimised, which includes minimising resource utilisation, cost and time.
- $DC_i(x)$: The i-th demand constraint function that must be satisfied by the resources x.
- $IC_j(x)$: The j-th investment constraint function that must be satisfied by the resources x.
- $SC_k(x)$: The k-th space constraint function that must be satisfied by the resources x.
- $TRA_l(x)$: The l-th total resource allocation constraint function that must be satisfied by the resources x.

- $SRR_m(x)$: The m-th specific resource requirements constraint function that must be satisfied by the resources x.
- m_1: The number of demand constraints.
- m_2: The number of investment constraints.
- m_3: The number of space constraints.
- n_1: The number of total resource allocation constraints.
- n_2: The number of specific resource requirements constraints.

By solving this optimisation problem, the optimal resource selection for the manufacturing system reconfiguration can be determined, considering the objectives and constraints. Resource reconfiguration can be achieved by adjusting the decision variables (i.e., the types and quantities of resources) of the assets within the manufacturing system. This process allows the system to adapt efficiently to different product families or operational requirements while minimising resource utilisation, cost, and time.

4.2.2.9.3 Job Scheduling In the proposed ontology model, the scheduling problem is described as a set of decisions concerning the sequence of parts to be released into the system, the selection of the operation/resource pair, and the sequence of parts assigned to each resource in the production process. This model includes the following information as subclasses of the decision variables:

- **Decision Variables:** Available assets to perform the manufacturing jobs, jobs that must be performed, and a set of operations for all jobs, which must be performed in a specific order based on the constraints.
- **Optimisation Objectives:** Makespan, the workload of the most loaded resource, production rate, flow time, tardiness, and resource utilisation.
- **Constraints:** In the context of job scheduling, constraints can be categorised into equality and inequality constraints:

    1. Inequality Constraints

(a) Shortest Processing Time (SPT): The job with the shortest processing time should be processed first.

(b) First In, First Out (FIFO): The job that entered the system first should be processed first.

(c) Most Work Remaining (MWR): The job with the most work remaining should be processed first.

(d) Earliest Due Date (EDD): The job with the earliest due date should be processed first.

2. Equality Constraints

(a) Machine Constraints (MC): A job can only be processed on one machine at a time.

(b) Job Constraints (JC): A job must be completed before the next job can commence.

(c) Asset Constraints (AC): An asset can only be used by one job at a time.

(d) Precedence Constraints (PC): Certain jobs may have a specific order in which they must be processed.

For example, if the objective is to optimise job scheduling for reconfiguration, then the following objectives must be considered:

1. Minimise Makespan (M)
2. Minimise Workload of the Most Loaded Asset (WL)
3. Minimise Tardiness (T)

The decision variables in this problem are the sequence of parts to be released into the system, the selection of the operation/asset pair, and the sequence of processes assigned to each asset

in the production process. The optimisation problem can be formulated as follows:

$$\begin{aligned} \text{Minimise} \quad & f(x) = \{M(x), WL(x), T(x)\} \\ \text{subject to} \quad & EDD_i(x) \leq 0, \quad i = 1, \ldots, m_1 \\ & MC_j(x) = 0, \quad j = 1, \ldots, n_1 \\ & JC_k(x) = 0, \quad k = 1, \ldots, n_2 \\ & AC_l(x) = 0, \quad l = 1, \ldots, n_3 \\ & PC_m(x) = 0, \quad m = 1, \ldots, n_4 \end{aligned} \tag{4.3}$$

where:

- x: A vector representing the decision variables.
- $f(x)$: A vector containing the objective functions to be minimised, which includes minimising makespan, the workload of the most loaded asset and tardiness.
- $\text{EDD}_i(x)$: The i-th earliest due date constraint function that must be satisfied by the decision variables x.
- $\text{MC}_j(x)$: The j-th machine constraint function that must be satisfied by the decision variables x.
- $\text{JC}_k(x)$: The k-th job constraint function that must be satisfied by the decision variables x.
- $\text{AC}_l(x)$: The l-th asset constraint function that must be satisfied by the decision variables x.
- $\text{PC}_m(x)$: The m-th precedence constraint function that must be satisfied by the decision variables x.
- m_1: The number of earliest due date constraints.
- n_1: The number of machine constraints.
- n_2: The number of job constraints.
- n_3: The number of asset constraints.

- n_4: The number of precedence constraints.

By solving this optimisation problem, the optimal solution for the job scheduling problem can be found, considering the objectives and constraints. The decision variables x represent the allocation of jobs to machines, and the constraints ensure that the schedule satisfies the earliest due dates, machine capacities, job requirements, asset availability, and job dependencies. The optimal solution helps to minimise the makespan, balance the workload, and reduce the total time needed to complete the job schedule.

4.2.2.10 Detailed Information about Robot in the Asset Semantic Model

Within the scope of this PhD research, understanding the robot's representation in the asset semantic model is crucial due to the focus on robotic assembly. The "Robot" class, a subclass of the asset semantic model, encapsulates critical attributes that define its primary characteristics. While these attributes are flexible and can be expanded upon, core attributes such as robot reachability, robot payload, and robot DH parameters are integral to the robot semantic model. This model is illustrated in Figure 4.5.

A distinction must be made between attributes and classes, with the latter encompassing more intricate details. For instance, the DH parameters are modelled as a class due to their variable nature across robot types.

4.2.2.10.1 Robot Reachability Reachability in robotics indicates the area a robot arm can cover from a static pose. Multiple factors influence reachability, including the robot's arm length, joint configuration, and orientation. This attribute is pivotal when considering the optimal pose of the robot within a manufacturing layout, encompassing both the position and orientation of the robot to maximise its operational efficiency and coverage.

4.2.2.10.2 Robot Repeatability Repeatability pertains to a robot's consistency in returning to a programmed position. This attribute is significant for ensuring product quality and operational efficiency in manufacturing.

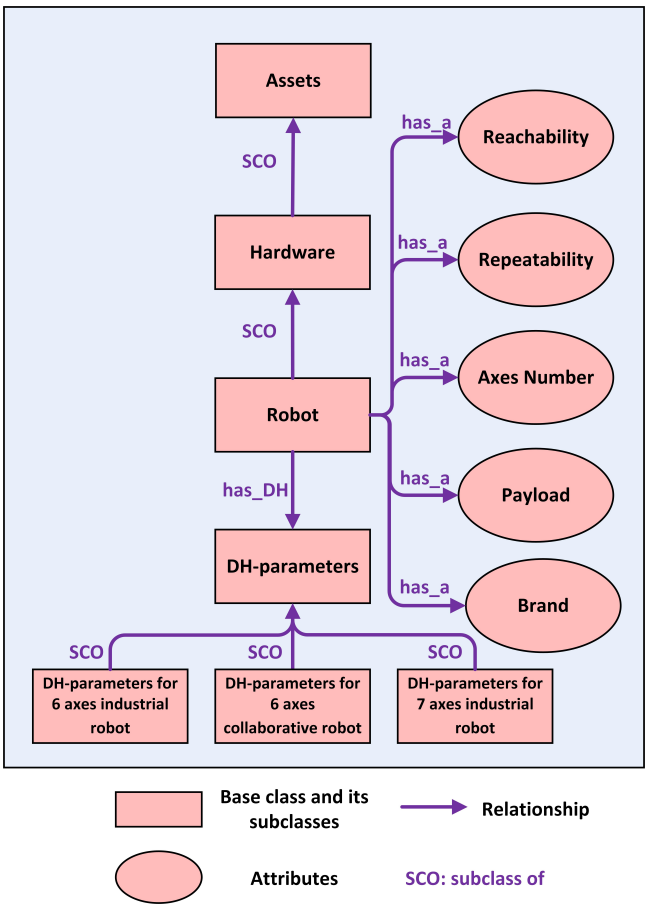


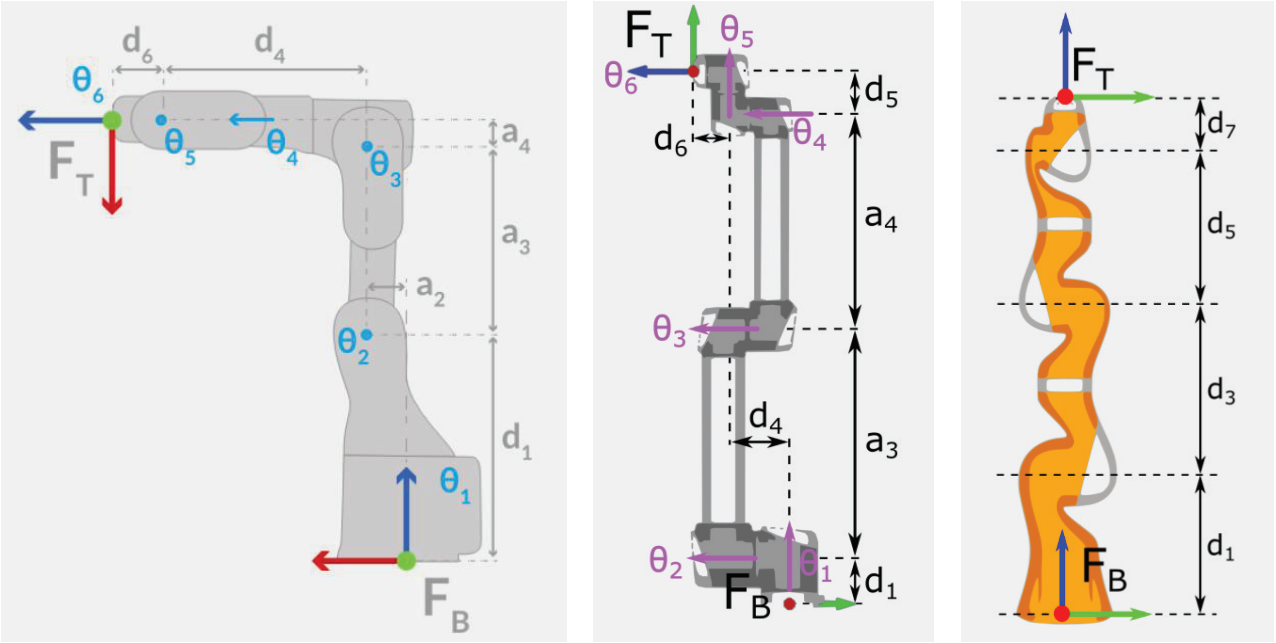
Figure 4.5: Robot semantic model

4.2.2.10.3 Robot Axes A robot's ability to move is based on its axes, similar to how our joints allow us to perform various movements. The more axes a robot has, the more versatile its movements can be. Robots can be designed in different configurations based on these axes. For example, some might move like a human arm, while others might resemble a spider or crane. Each configuration is suited for specific tasks.

4.2.2.10.4 Robot Payload The payload in robotics refers to the maximum weight a robot can handle, excluding its own weight. It is a straightforward yet vital attribute ensuring that the robot can efficiently handle the assigned tasks.

4.2.2.10.5 Robot DH Parameters The breadth of robotics applicability in this research necessitates understanding each robot's unique characteristics, particularly the DH parameters, which have been thoroughly reviewed in previous literature. Additionally, the dimensions and size of the end effector have been captured within the semantic model. When a specific robot and its corresponding end effector are chosen, the pose of the end effector can be determined.

This process involves standardising the robot structure and integrating all pertinent information into the knowledge graph. In the experience databank, the robots are categorised into three distinct types: six-axis industrial robots, six-axis collaborative robots, and seven-axis industrial robots. This classification captures a broad range of robots utilised in manufacturing-related robotic assembly operations. Despite each robot type employing a unique method to calculate the end effector's pose, the required parameters remain consistent across types. Provided the technical specifications, these parameters can be documented as shown in Figure 4.6.



a. 6 axes industrial robot b. 6 axes collaborative robot c. 7 axes industrial robot

Figure 4.6: Required parameters for determining the end effector pose

As depicted in Figure 4.6, once the relevant parameters unique to the type of robot are understood, the pose of the robot's end effector can be calculated accordingly based on the DH parameters and joint angles [99]. For instance, with a six-axis industrial robot, knowledge of the dimensions θ_1, θ_2, θ_3, θ_4, θ_5, θ_6, d_1, d_4, d_6, a_2, a_3, a_4 facilitates the computation of the end effector pose. This information can be conveniently sourced from the technical documentation provided with each robot.

The DH parameters are crucial for layout optimisation. They provide a systematic method to compute the pose of the robot's end effector. Thus, when the robot changes position, the DH parameters can accurately calculate the new pose of the end effector, thereby ensuring optimal operation and efficiency of the robot. However, when these DH parameters are employed, the definitions of the end effector's pose, its calculation, and the movement of the robot must be

considered, all of which can vary significantly across different robot brands. Different robots and software may use unique conventions for rotation sequences, and understanding the specific convention in use is vital. Furthermore, the orientation of the end effector's pose can also vary, as different Euler angles have different representations. This information is included in the experience database for future reference. While some systems use extrinsic orientation, others use intrinsic orientation. In extrinsic rotation, the object is turned based on a static, unchanged coordinate system, whereas with intrinsic rotation, the object spins based on its own coordinate system, which evolves with each turn. This distinction is crucial in robotics, as it directly influences how the robot interprets and executes movement commands. As illustrated in Figure 4.7, different robot brands possess unique orientation representations, and recognising these differences is pivotal for improving the accuracy of the work.

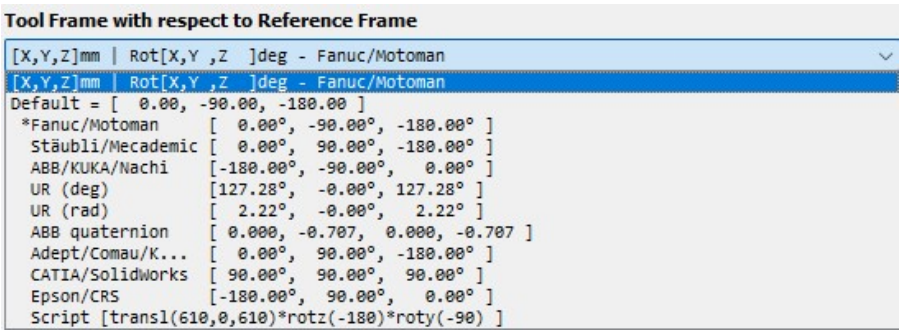


Figure 4.7: Orientation definition based on different brands

Euler angles offer a method to represent the orientation of a coordinate frame through a series of three rotations around distinct axes. The symbols ' and " indicate that the rotations are performed about the newly transformed axes rather than the original axes.

The ABB notation [Z, Y', X"] demonstrates rotations occurring within the coordinate system defined by the previous rotation, reflecting the intrinsic nature of Euler rotations.

In contrast, the FANUC notation Rot[X, Y, Z] deg suggests that rotations occur around the original fixed axes, denoting extrinsic rotations.

While understanding FANUC's rotation concept is important, comprehending the interaction between its Joints 2 and 3 is equally crucial. Unlike in many robots where the rotation of Joint 2 affects that of Joint 3, in FANUC systems, Joint 3 rotates specifically in relation to the robot's base frame, not in relation to Joint 2. As shown in Figure 4.8, the calculation method for the end-effector pose of a FANUC robot differs from that of other robots. This ensures that Joint 3's reference remains consistent with the base, irrespective of Joint 2's movement. To illustrate,

if Joint 2 turns by 10 degrees and Joint 3 by 15 degrees about the base, the total effective rotation is 25 degrees. This unique interplay between Joint 2 and Joint 3 is fundamental to FANUC's design and functionality. A deep comprehension of this interaction is essential for those programming and operating these robots. Moreover, the experience databank stores this information for layout optimisation purposes to ensure accurate derivation of the DH parameters for FANUC robots.

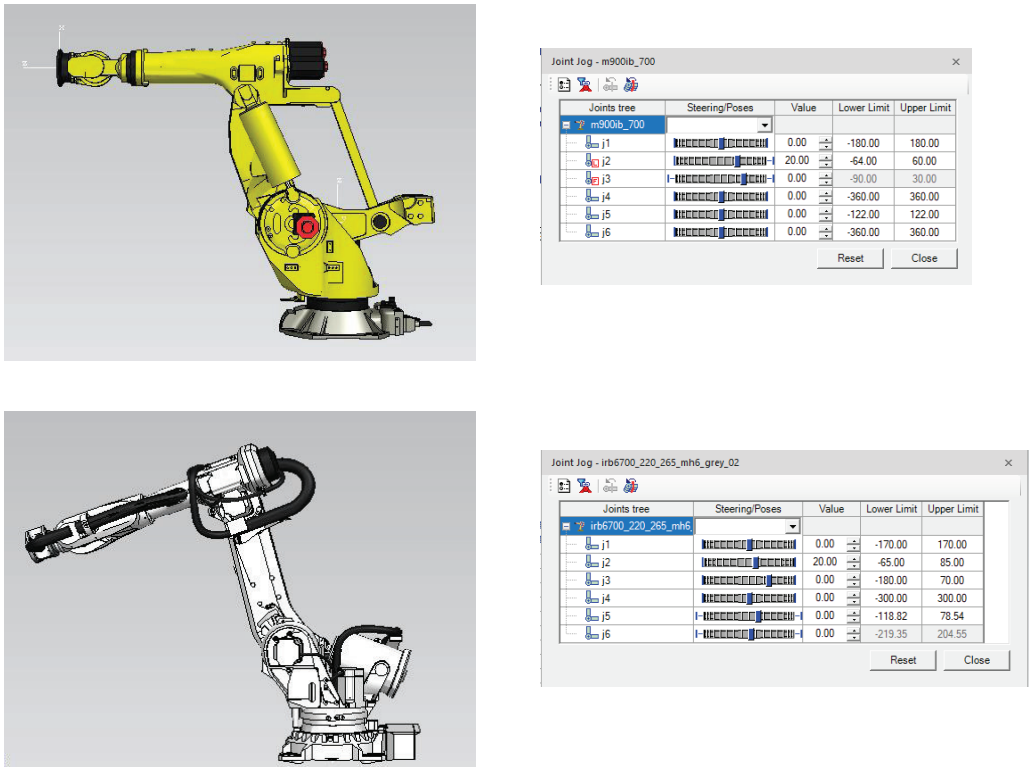


Figure 4.8: FANUC J2 and J3 interaction

Considering these elements, the general dimensions of the DH parameters have been defined in the ontology model, together with brand-specific information. Even though the proposed ontology model aims to be vendor-neutral, the unique operational concepts of individual robots must inevitably be considered in real-world applications, as different brands adhere to distinct concepts.

4.2.3 Construction of the Entity Layer of the Knowledge Graph

As mentioned in Section 2.3, the knowledge graph consists of a schema layer and an entity layer. Following the description of the schema layer, the approach to building the entity layer is now discussed in this section. The creation of the entity layer of the knowledge graph is

based on the initial schema layer, which, in this case, is the OCCR model. The schema layer is then continually refined through the integration of bottom-up and top-down methods in the proposed framework by incorporating valuable information and insights obtained from the entity layer. Figure 4.9 presents the detailed steps for building the entity layer of the knowledge graph.

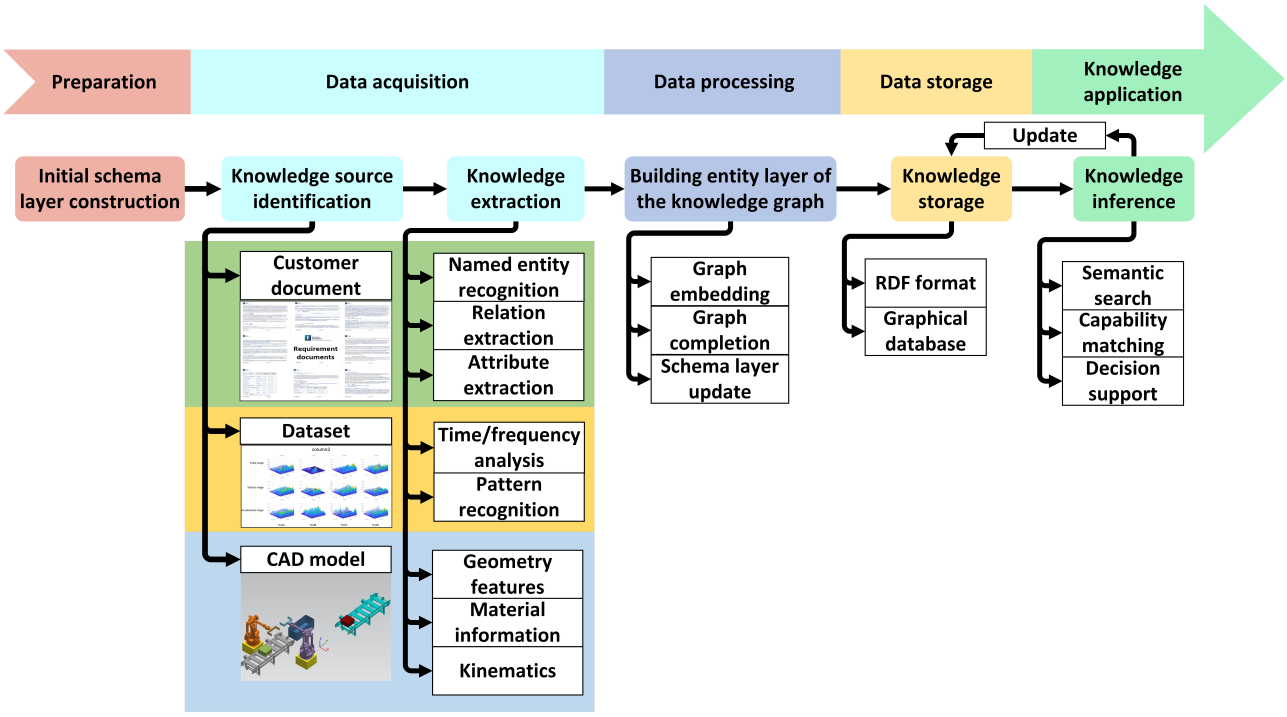


Figure 4.9: Process of construction of the entity layer of the knowledge graph

The implementation of each step will depend on the application domain and organisation. Information about each step is detailed below:

1. **Initial Ontology Construction**

   The proposed methodology uses the OCCR model as the initial ontology model.

2. **Knowledge Source Identification**

   The next step is to identify the different sources of manufacturing data, which will be used to tailor the model to the application domain. In manufacturing scenarios, the knowledge resource usually comprises heterogeneous data sources, including customer requirement documents, datasets, and Computer-Aided Design (CAD) models. These resources are multi-modal, with different forms, and hence require separate processing methods.

3. **Knowledge Extraction**

   The knowledge extraction method is applied to extract the source data. For the cus-

tomer requirement document, natural language processing techniques such as named entity recognition [100], relation extraction [101], and attribute extraction [102] could be utilised. The knowledge extraction process combines the manufacturing domain knowledge and terms as the keyword corpus. The time/frequency analysis and pattern recognition process are applied to extract the data for the dataset. For the CAD model data, a data extraction tool such as API is applied to extract the essential geometric features, material information, and kinematics [103].

4. **Building the Entity Layer of the Knowledge Graph**

   The entity layer of the knowledge graph is established using the extracted data and the ontology model described in Section 4.2.2 to construct the schema layer. Despite being constructed from multiple sources, the generated knowledge graph may still have incomplete information – specifically, it may be missing certain triples. To compensate for this, the knowledge graph undergoes a combination of completion steps: manual completion by engineers, completion based on established rules, and automatic completion using either graph structure or embedding-based algorithms [104].

5. **Knowledge Storage**

   Once the entity layer of the knowledge graph is established and updated, it can be stored in either graph databases [105] or RDF [97] format. These storage solutions provide efficient querying capabilities and effective management of substantial volumes of knowledge graph data.

4.2.4 Application of the Generated Knowledge Graph (Experience Databank)

After the entity layer of the knowledge graph is generated, the experience databank in the robotic assembly cell is achieved and can be applied to various aspects of RMSs and beyond. The generated schema layers enable reasoning for capability matching and recommendations for reconfiguration solutions, as illustrated in Figure 4.10. Capability matching should consider not only the static requirements, such as the required payload and reachability, but also the capacity information, such as the cost and utilisation rate. If there are multiple candidate assets after the capability-matching process, then they are evaluated before an appropriate asset is

selected, as explained in Section 5.4. Once the reconfiguration model recommends a solution, the engineer will decide on the criteria for optimising the reconfiguration. Layout optimisation, resource selection, and job scheduling are typically multi-objective optimisation problems. If the engineers can decide the objectives' weights in advance, then the multi-objective optimisation problem can be converted into a single-objective optimisation problem [106]. Otherwise, a posteriori method can be used to produce all Pareto-optimal solutions or a representative subset of those solutions [107].

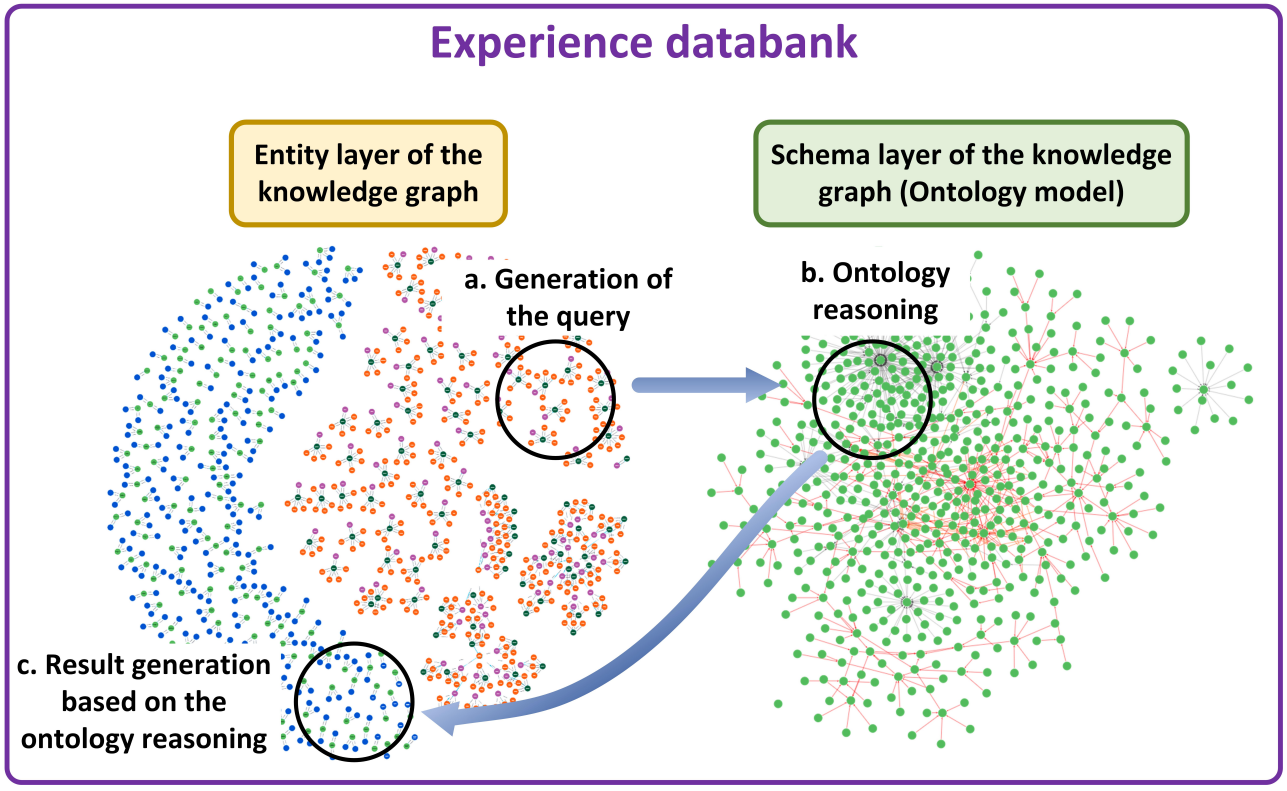


Figure 4.10: Knowledge inference process in the experience databank (achieved by knowledge graph): a. Generation of the query. b. Ontology reasoning. c. Generation of the inference results based on the ontology reasoning

4.3 Chapter Summary

This chapter delves into the resolution of the first research question posed in this PhD study, specifically RQ1: "How can data from various sources be efficiently processed by integrating diverse systems and technologies into robotic assembly cells?" A formal and unified ontology model and a knowledge graph-based method are introduced to address this. While principally designed for data storage and representation, the ontology model represents information's ca-

pabilities, capacities, and reconfiguration in a vendor-neutral manner. Through this representation, it aids in the interpretation of data by providing structured insights. The process efficiency, however, is chiefly derived from the knowledge graph-based approach, which harmoniously blends top-down and bottom-up methodologies for the construction and periodic updates of the knowledge graph. This approach not only ensures accurate data representation but also streamlines its processing. The experience databank is established via the knowledge graph.

This chapter emphasises the synergy of the ontology model and the knowledge graph in efficient processing and interpreting robotic assembly data. While the ontology model provides a robust foundation, the knowledge graph-based method actualises the objectives of efficient processing and interpretation, aligning with Objective 1.

Chapter 5

Optimal Manufacturing Asset Selection

This chapter is designed to address RQ2 and fulfil Objective 2. A methodology that effectively identifies suitable assets using their specifications and production capacity, known as "candidate asset identification", is proposed. Furthermore, a methodology for evaluating these assets, considering their specifications and capacity, called "candidate asset evaluation", is proposed. The overarching aim is to ensure that robotic assembly cells can adapt swiftly and appropriately to any changes in their operational requirements.

5.1 Introduction

In the process of candidate asset identification, two main requirements must be fulfilled. First, the methodology should enable a swift and effective understanding of and response to new process requirements. This involves a thorough comprehension of the new requirements. After adjusting to the new process requirement, a bill of process is created. Second, a capability assessment procedure must be executed to facilitate matching capabilities. These two requirements are essential for the identification of candidate assets.

Given the variety of methods available for asset evaluation, the modularity of the methodology for candidate asset evaluation must be validated, as modularity increases the flexibility and scalability of the framework. In addition, the methodology should possess characteristics of MCDM due to the numerous criteria involved in asset evaluation. Such characteristics greatly enhance the evaluation process. Finally, the design of the evaluation methodology should

incorporate recommendations from the experience databank, as suggested in Chapter 4. This aids in simplifying the candidate asset evaluation process.

5.2 Methodology for Optimal Manufacturing Asset Selection

In the implemented framework, the experience databank established in Chapter 4 is used for asset selection and will determine the assets required to enable the creation of the product (or will lead to the decision that creation of the product is not possible). At this stage, the product requirements in the form of the bill of process have been decomposed into sub-product requirements. The core procedures for this decomposition include rule-based, semantic-based, and semantic-embedding-based methods, which are performed in the product requirements step of the framework. This methodology will begin with a search of the experience databank for any existing bill of resources that could be used for the whole product or sub-product requirements.

If no suitable bill of resources is available or if gaps exist, the experience databank is utilised to conduct capability matching between the current system configuration and the required processes to determine the bill of resources. For capability matching, the entity representing the required system configuration queries the current system configuration entities to check whether capabilities fulfilling the product requirement are available. If multiple assets meet the capability requirement simultaneously, the next step is to select the most suitable candidate assets. Figure 5.1 illustrates the simplified asset selection process.

Algorithm 1 provide detailed information about the asset selection process using the evaluation method and the experience databank. The final decisions can be as follows:

1. The current system can produce the product. In this case, either one or multiple combinations of the assets can be selected during the capability-matching process. If more than one combination can be applied to produce the product, the most suitable assets will be selected (in other words, asset evaluation will occur).

2. The current system is unable to produce the product. In this case, the decisions will add

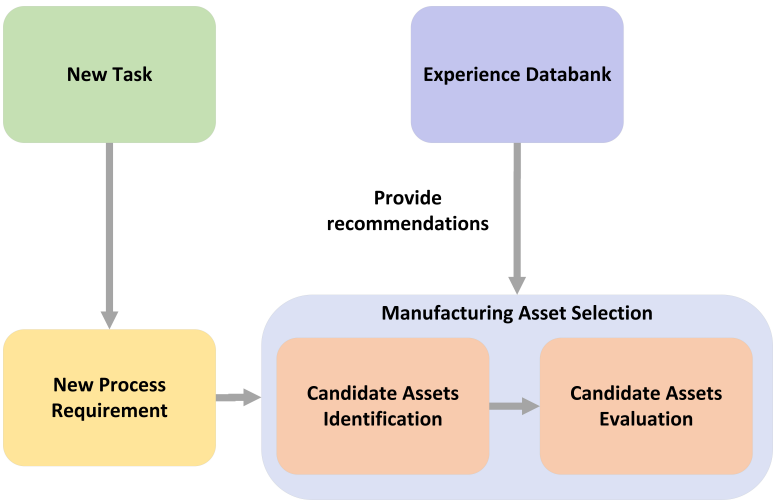


Figure 5.1: Simplified manufacturing asset selection process.

or update assets to match the product requirement. Information about other potential assets to be added to the experience databank will be suggested to the user.

In summary, two steps are executed in the candidate asset selection process. The first step is to identify the candidate assets for the new product and the new process requirements. If more than one candidate asset is found, then the assets are evaluated in the second step, and those with the highest scores are selected.

5.3 Methodology for Identifying Candidate Assets

This section explains the methodology for identifying the candidate assets. Figure 5.2 illustrates the workflow of this methodology, which includes candidate asset identification with static information and results enhancement with capacity information.

The capability-matching process in the robotic assembly cell should be executed to complete the candidate asset identification process. Capability matching involves aligning a process's requirements with the assets' capabilities and is fundamental for identifying appropriate candidate assets. This comprehensive and logical methodology is key to identifying the optimal capacity assets to meet specific process requirements. As implemented in Figure 4.10 of Chapter 4, ontology reasoning is utilised for capability matching.

To accomplish effective capability matching, the requirements must be considered. In the

Algorithm 1 Methodology for Asset Selection

Require: Initialised knowledge graph as *experience databank*, New Task as NT
Ensure: Decisions at the reconfiguration stage 1 as D

Load *experience databank* and NT
ProductRequirement (PR)=$Find\_requirement(NT)$
SubProductRequirement (SPR)=$Divide\_ProductRequirement(PR)$
Initialise bill of process set ($BoPs$)
for each $SPR \in PR$ **do**
    BillofProcess (BoP)=$Process\_search\_rule(SPR)$
    if BoP ! = $Null$ **then**
        $BoPs.add(BoP)$
        Continue
    else $BoP = Process\_search\_semantic(SPR)$
        if BoP ! = $Null$ **then**
            $BoPs.add(BoP)$
            Continue
        else $BoP = Process\_search\_semanticEmbedding(SPR)$
            $BoPs.add(BoP)$
            Continue
        end if
    end if
end for
Initialise decisions (D) and score set (ScS)
for each $Process \in BoPs$ **do**
    Required system configuration (RSC) = $Find\_required\_configuration(Process)$
    Resources (Rs) = $Matching\_capability(RSC, experience\ databank)$
    if Rs ! = $Null$ **then**
        for each $R_i \in R_s$ **do**
            Score (Sc) = $Evaluation(R_i)$
            $ScS.add(Sc)$
        end for
        $R_h = assetSelection(ScS)$, ▷ where R_h has the highest score
        $D.add(R_h)$
        Continue
    else $D.add(ReconfigurationSuggestions)$
    end if
end for
return D

robotic assembly process, different types of assets have different requirements – not only static requirements but also capacity-related information. Static requirements refer to the unchanging attributes needed for a task, such as essential payload and reachability. This methodology primarily focuses on employing stringent filtering criteria, often referred to as hard Boolean limits, to sieve through potential assets. To elaborate, if a particular process requires a robot to have reachability beyond 2,000 millimetres, only those that meet or exceed this benchmark are considered; those falling short are instantly ruled out.

Capacity information should also be considered. In robotic assembly cells, capacity information such as utilisation rate and cost should be considered as extra information to enhance the results of candidate asset identification.

This approach refines the selection process by incorporating both static specifications and capacity criteria. Only assets that satisfy all the critical benchmarks are considered viable candidates for the task. The employed method ensures that the selected assets are theoretically capable (based on their specifications) and practically feasible, considering aspects such as cost

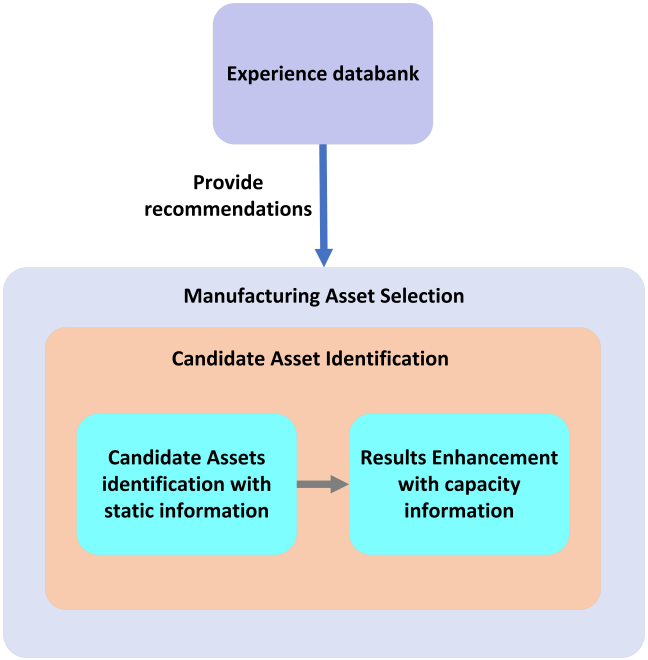


Figure 5.2: Workflow of identifying the candidate assets

and utilisation frequency, thereby facilitating an optimal and logical asset selection process. This information is from the proposed ontology model.

5.3.1 Identifying Candidate Assets Based on Static Information

Identifying candidate assets based on static information is important, as it determines whether the potential assets can be used to execute the required operations. For a clear and structured overview, Table 5.1 presents a selection of common static specification criteria for asset identification in robotic manufacturing cells, as proposed by the generated experience databank. While not exhaustive, this list serves as a representative example of criteria for specific asset types.

The table demonstrates that various criteria, including the number of robot axes, reachability, repeatability, and payload, are used to select robots, as outlined in the experience databank. These criteria can be tailored to align with customer needs and specifications. These specifications must align with the requirements of the task to guarantee that the robot operates efficiently within the assembly cell.

Beyond robots, finger grippers play a critical role in robotic assembly. As the end effector of the robot, finger grippers can be used for pick-and-place tasks. The maximum and minimum

Table 5.1: Examples of criteria for selecting assets

Asset Type	**Specification Criteria**
Robot	Required payload, required number of robot axes, required reachability, required repeatability
Finger Gripper	Allowed maximum grasping force, allowed minimum grasping force, required number of fingers
Vacuum Gripper	Allowed maximum grasping force, allowed minimum grasping force, required number of cups
Metrology Device	Required measurement frequency, required lateral offset, required planar offset, required radial offset
Marker	Required accuracy, required resolution
Drilling Tool	Allowed maximum drilling depth, allowed maximum hole diameter, allowed minimum hole diameter

allowable grasping force and the required number of fingers are important considerations, as they dictate the types of objects the gripper can handle. For instance, a higher grasping force allows the gripper to handle heavier objects, while more fingers can provide better grip and stability, especially for irregularly shaped objects. Furthermore, the minimum grasping force is necessary to handle fragile objects without damaging them. Therefore, these specifications must be considered carefully to ensure effective and safe operation.

A vacuum gripper, similar to a finger gripper, is another type of end effector used for pick-and-place tasks. It shares some specification criteria with the finger gripper, such as the maximum and minimum allowable grasping force. However, it also has a unique criterion, namely the required number of cups. This number is important because it directly influences the surface area that the gripper can adhere to. A greater number of cups allows the vacuum gripper to handle larger or heavier objects, as the suction force is distributed over a larger surface area. Additionally, multiple cups provide redundancy: if one cup fails or cannot establish a secure seal due to the object's surface irregularities, the others can still maintain a firm hold.

A metrology device is yet another vital asset in a robotic assembly cell. As a measurement tool, it is responsible for ensuring the accuracy and precision of the manufacturing process. The specific criteria for metrology devices in this study's model include the required measurement frequency, required lateral offset, required planar offset, and required radial offset. The required measurement frequency refers to how often the metrology device takes measurements. This is important because frequent measurements can aid in the early detection of errors or deviations in the manufacturing process, allowing for timely corrections and ensuring the quality of the final product. The required lateral, planar, and radial offsets are critical for determining the

allowable deviation or error in the measurements the metrology device takes. Lateral offset refers to the allowed side-to-side deviation, while planar offset refers to the allowed deviation in a flat plane, and radial offset refers to the allowed deviation in a circular path or radius around a certain point. These offsets are vital because they set tolerance limits for the manufacturing process. Exceeding these limits could lead to defective products. Therefore, these offsets must be carefully set according to the precision required for the specific application.

The marker end effector is another crucial component in the robotic assembly cell, particularly for marking operations. The key specifications for a marker include the required accuracy and resolution. The required accuracy pertains to the marker's ability to place marks at precise locations as the manufacturing process dictates. Any inaccuracy can lead to misinformation or misinterpretation of the marked data, impacting the overall quality and traceability of the product. The required resolution refers to the smallest possible mark the end effector can make. A higher resolution allows for more detailed and intricate markings, which can be particularly important for products that require detailed instructions, identifiers, or aesthetic designs to be marked directly onto them.

Similarly, the drilling tool is also an end effector that plays a significant role in the robotic assembly cell. Drilling is a standard operation in many manufacturing processes, making the drilling tool's capabilities crucial. The maximum allowable drilling depth determines the depth to which the drilling tool can penetrate the material, which is essential when creating holes for fasteners or other components. The maximum and minimum hole diameters dictate the size of the holes that the drilling tool can make. These values are vital to ensuring that the holes are suitable for their intended purpose, such as accommodating specific fasteners or allowing certain tolerances. Through careful consideration of these criteria, the right drilling tool can be selected, thus ensuring effective and efficient operation within the robotic assembly cell.

5.3.2 Enhancing the Result Based on the Capacity Information

Capacity information plays a vital role in refining the identifying candidate assets. This information extends beyond the static specifications of an asset, providing insights into the dynamic, operational attributes that can impact an asset's feasibility for a task. As mentioned in Section 4.2.2, the capacity information is stored in the OCCR model and experience databank. It can

be used to enhance the result of candidate asset identification.

Cost is a key element of capacity information. Even if an asset technically meets all the specification criteria required for a task, if its cost exceeds the budget allocation for the project, it becomes an impractical choice. Therefore, the financial aspect is a crucial consideration in the selection process.

Availability is another essential component of capacity information. An asset could meet all the specifications and be within the budget, but if it is not available when required, it cannot be selected as a candidate. Reasons for unavailability could range from maintenance schedules and previous commitments to supply chain issues.

The utilisation rate, indicating the proportion of time an asset is in use compared with its total available time, is also a key aspect of capacity information. An asset with a high utilisation rate may be less available for new tasks, and its frequent use could lead to higher maintenance needs. Conversely, an asset with a lower utilisation rate may be more readily available and potentially have lower maintenance needs due to less wear and tear.

By considering capacity information alongside static specifications, the selection process can be significantly enhanced. This approach ensures that the chosen assets are not only technically capable of performing the task but also practically feasible, considering factors such as cost, availability, utilisation rate, and setup time. This comprehensive approach ultimately leads to a more effective and efficient asset selection process.

5.4 Methodology for Evaluating Candidate Assets

5.4.1 Introduction

After the initial identification of candidate assets, it is plausible that multiple assets may fulfil the established specification and capacity criteria. These candidate assets must be evaluated to find the optimal selection. This section describes the methodology employed for evaluating candidate assets, ensuring that the chosen assets meet the specifications but are also the best options within the given constraints. In the implemented framework, asset evaluation entails an analysis of equipment and technology to enable efficient production. This process requires

careful consideration of multiple factors, including the asset's specification information, such as a robot's reachability and repeatability, and capacity information, such as cost and resource utilisation. Traditionally, an engineer manually evaluates the assets. By contrast, the proposed framework employs evaluators, or evaluation metrics, to inform decisions about asset selection. These evaluators are not personnel but quantitative or qualitative measures used to assess the suitability and effectiveness of assets. Examples of evaluators include cost analysis, efficiency ratings, and reliability measures.

The evaluators play a vital role in asset evaluation in the manufacturing industry. These metrics guide the optimal choice of equipment and technology, which is important for effective and efficient production processes. Within the implemented framework, evaluators incorporate specification information from the capability model and capacity information, as detailed in Chapter 4. The following section details some of the commonly used evaluators within this framework.

5.4.2 Evaluators Definition

A comprehensive array of evaluation metrics can be utilised in the proposed methodology. Specific metrics should be used within a robotic assembly cell, including the specification efficiency score and reconfiguration costs. The specification efficiency score, one of the primary evaluators, is derived by tallying scores from several subsidiary evaluators or specifications, such as reachability, repeatability, and payload. The reconfiguration cost, which includes purchase, installation, and removal costs, covers the critical cost of robotic assembly reconfiguration. These two primary evaluation metrics are crucial for directing an optimised asset selection process. This process is systematically structured to align with the specific objectives and constraints inherent in the robotic assembly cell.

For clarity, the specification efficiency score and reconfiguration costs should be categorised as primary evaluators due to their significant role in the methodology. Meanwhile, the metrics employed in the calculation of the specification efficiency score, including reachability, repeatability, and payload, can be labelled as subsidiary evaluators. This classification clarifies the different roles of these evaluators within the broader evaluation process.

5.4.2.1 Specification Efficiency Score

In today's dynamic and fiercely competitive market landscape, astute resource management emerges as a linchpin for sustainability and competitiveness. A critical component of this management is circumventing over-specification, which occurs when an asset boasts capabilities that are disproportionately superior compared with what is necessitated by a specific task. This incongruence culminates in inefficiencies and an imprudent expenditure of resources. The present section offers a discourse on over-specification within resource evaluation, focusing on robotic assembly cells, and underscores the imperative of harmonising asset capabilities with task stipulations.

Within robotic assembly cells, robots and other end effectors must be chosen carefully to avoid unnecessary costs and to boost production. When it comes to robots, over-specification can lead to higher capital costs and upkeep costs without optimal use of the robots' natural abilities. Moreover, choosing the right end effectors, such as grippers and vacuum cups, is crucial because they touch the workpiece directly and affect the system's accuracy and trustworthiness. Within robotic assembly cells, robots and other end effectors must be chosen carefully to avoid unnecessary costs and to boost production.

Addressing over-specification involves carefully analysing the differences between task requirements and the specifications of available resources. This process involves identifying key aspects such as payload, reachability, and precision and then calculating the difference between the task's needs and the resources' capabilities with regard to each aspect. Summing up these differences results in a specification efficiency score for each resource. This score is essential for finding a resource that matches the task's requirements without being unnecessarily overqualified.

The "specification efficiency score" is a custom metric that evaluates the alignment of a resource's features with the stipulated requirements. A high score indicates that the resource meets the requirements without superfluous elements. By contrast, a low score suggests that the resource has many specifications or features that extend beyond or are not aligned with the defined requirements. In essence, this score aids in determining how efficiently a resource adheres to the set specifications without unnecessary additions. Reducing over-specification is essential when allocating resources, particularly in robotic assembly cells. Companies can make

intelligent choices that optimise resource use and prevent unnecessary costs by meticulously comparing task requirements and resource specifications. The specification efficiency score is vital in this process, providing a measurable value for informed decision-making. Figure 5.3 presents a simple workflow for calculating the specification efficiency score using the developed framework.

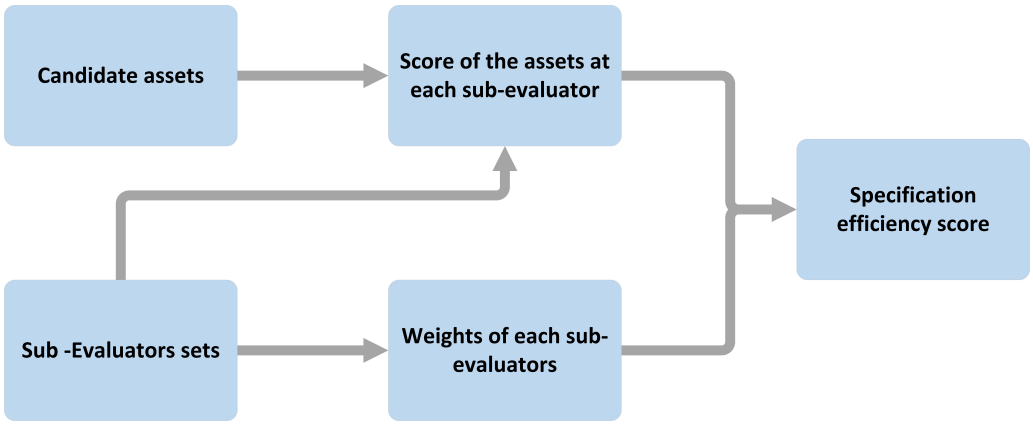


Figure 5.3: Simplified process for calculating the specification efficiency score

This figure illustrates how various sub-evaluators are used to assess candidate assets. The choice of sub-evaluators depends on the type of assets under consideration and the customer's specific needs, as detailed in Table 5.1. For instance, if multiple robots are being evaluated as potential assets, then reachability, repeatability, and payload can be used as the relevant sub-evaluators to calculate the specification efficiency score. First, the score for the candidate asset at each sub-evaluator is calculated. Then, the final specification efficiency score is determined by considering the weights assigned to each sub-evaluator. This constitutes a clear and straightforward method to compare candidate assets quantitatively, guiding selection towards the most suitable option. Figure 5.4 explains how the specification efficiency score is calculated in the proposed optimal asset evaluation methodology.

Consider a group of processes, denoted as $p_1, p_2, \ldots, p_j$. This group encompasses j distinct processes that are potential candidates for executing a specific task. In order to systematically define the candidate assets D for each process, m is introduced to represent the position of a process in the sequence. Here, m is an integer and belongs to the set [1, j]. Each process m corresponds to a total of k_m candidate assets. The set of candidate assets associated with process m can be mathematically expressed as shown in Equation (5.1). By adopting this method, a structured representation of the sets of candidate assets associated with each distinct

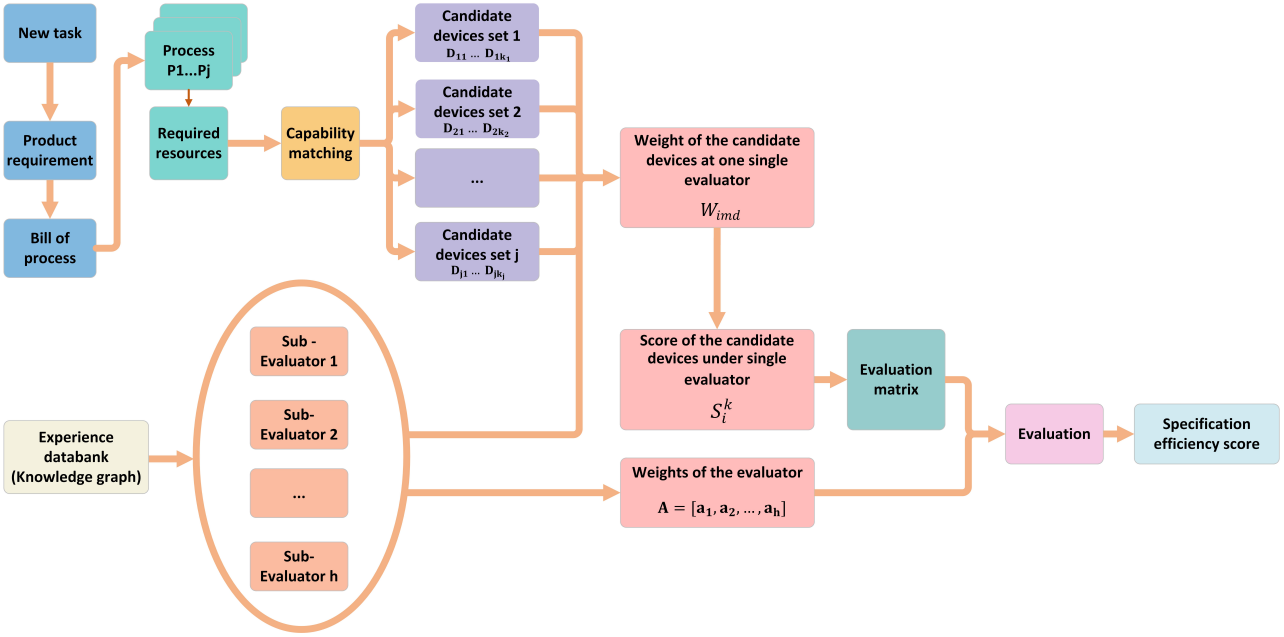


Figure 5.4: Detailed process of calculating the specification efficiency score

process can be achieved.

$$C_m = D_{m_1}, D_{m_2}, ..., D_{m_{k_m}}, \quad (\text{where } m \in [1, j] \cap \mathbb{Z}) \tag{5.1}$$

In the aforementioned equation, C_m signifies the group of candidate assets for process m. Additionally, D_{m_i} denotes the i-th potential asset that corresponds to the process m. This mathematical formulation allows for efficient grouping of potential assets for each considered process.

The weights for each candidate asset in every process, evaluated by a specific sub-evaluator, can be calculated as shown in Equation (5.2), where w_{imd} represents the weights of the d-th candidate asset set, evaluated by evaluator i at process m. h denotes the total number of sub-evaluators. The type of sub-evaluator differs based on the candidate asset sets under consideration. For instance, if the candidate assets are robots, then the distinguishing features or evaluators could include the type of gripper, capability model, reachability, and repeatability. These sub-evaluators are used in the proposed approach to calculate the specification efficiency score. Sub-evaluators can be represented as vectors [108, 109], or they can be defined numerically based on criteria designed by engineers drawing from their experience [110]. Finally, $\sigma_{md\_eva\_i}$ is the deviation between the i-th evaluator set and the corresponding features of the

d-th asset at process m.

$$w_{imd} = \frac{\sigma_{md\_eva\_i}}{\sum_{n=1}^{k_m} \sigma_{mn\_eva\_i}},$$
$$(\text{where } d = [1, k_m] \cap \mathbb{Z}, k_m = [k_1, k_j] \cap \mathbb{Z}, m = [1, j] \cap \mathbb{Z}, i = [1, h] \cap \mathbb{Z}) \tag{5.2}$$

In the evaluation process for candidate assets, an essential metric is the deviation between the property of a candidate asset and the ideal property defined by the sub-evaluator set. This deviation quantifies the difference between the actual property value of a candidate asset and the desired property value as specified by the sub-evaluator.

To illustrate, if the required reachability (a sub-evaluator) is 2,000 mm and a candidate asset (e.g., Robot 1) has a reachability of 2,500 mm, then the deviation amounts to 500 mm. Similarly, for another candidate asset (e.g., Robot 2) with a reachability of 3,000 mm, the deviation would be 1,000 mm. Here, a smaller deviation implies a better match between the candidate asset's properties and the desired properties defined by the sub-evaluator set. Candidate assets with smaller deviations —indicating a closer alignment with the sub-evaluator set — should consequently be assigned lower weights.

However, given that the most suitable candidate asset is identified based on the highest score according to Algorithm 1, using the reciprocal of the weights, w_{imd}, for further evaluation makes more sense. This allows assets with smaller deviations (and therefore closer matches) to have higher scores. Therefore, the updated weights, W_{imd}, can be calculated as follows:

$$W_{imd} = \frac{1}{w_{imd}} = \frac{\sum_{n=1}^{k_m} \sigma_{mn_eva_i}}{\sigma_{md_eva_i}},$$
$$(\text{where } d \in 1, \ldots, k_m, k_m \in k_1, \ldots, k_j, m \in 1, \ldots, j, i \in 1, \ldots, h). \tag{5.3}$$

Furthermore, the set of updated weights for the asset numbered as d in process m, denoted by W_{md}, can be expressed as a collection of weights across all sub-evaluators. This set is represented as:

$$W_{md} = [W_{1md}, \ldots, W_{hmd}]. \tag{5.4}$$

This formalism systematically captures the evaluation criteria and provides a structured approach to assessing and selecting candidate assets based on defined metrics.

Take process p_1 as an example; there are a total of k_1 candidate sets, and the weights of the candidate sets will be calculated. W_{111} describes the weights of sub-evaluator 1 of the candidate asset D_{11} at Process 1, and $\sigma_{11\_eva\_1}$ is the deviation between Evaluator Set 1 and the asset accuracy-related information of the above-mentioned asset.

After the weights of different assets at each process are calculated, the scores S_d^i of the candidate assets under the sub-evaluator U_i should be calculated, where $d = [1, k_m] \cap \mathbb{Z}$ and $i = [1, h] \cap \mathbb{Z}$. This score indicates the candidate assets' matching ability under a single sub-evaluator. The score can be defined in numerous ways as long as it conveys the matching ability of the candidate assets.

Taking the consistent fuzzy matrix as an example, the scores can be calculated as follows [111]. Considering that the most suitable assets from k_m candidate assets under h sub-evaluators must be selected, $h \cap \mathbb{Z}$ of single-evaluator fuzzy priority relations can be built.

$$B_i = (b_{ef}^i)_{k_m \times k_m}, (\text{where } k_m = [k_1, k_j] \cap \mathbb{Z}, i = [1, h] \cap \mathbb{Z}) \tag{5.5}$$

where b_{ef}^i is called the coefficient of the preferred relationship of asset D_{me} to asset D_{mf} under the sub-evaluator U_i. The value of b_{ef}^i is listed in (5.6).

$$b_{ef}^i = \begin{cases} 0, & \text{if } D_{me} \text{ is worse than } D_{mf} \text{ under the factor } U_i \\ 0.5, & \text{if } D_{me} \text{ is equal to } D_{mf} \text{ under the factor } U_i \\ 1, & \text{if } D_{me} \text{ is better than } D_{mf} \text{ under the factor } U_i \end{cases} \tag{5.6}$$

Then $B_i (i = [1, h] \cap \mathbb{Z})$ will be converted to the consistent fuzzy matrix as listed in (5.7) below.

$$R_i = (r_{ef}^i)_{k_m \times k_m} \tag{5.7}$$

where

$$r_{ef}^i = \frac{r_e^i - r_f^i}{2k_m} + 0.5; r_e^i = \sum_{l=1}^{k_m} b_{el}^i \tag{5.8}$$

Because for $\forall l = 1, 2, \ldots, k_m$,

$$r_{el}^i - r_{fl}^i + 0.5 = \frac{r_e^i - r_l^i}{2k_m} + 0.5 - (\frac{r_f^i - r_l^i}{2k_m} + 0.5) + 0.5 = \frac{r_e^i - r_f^i}{2k_m} + 0.5 = r_{ef}^i \tag{5.9}$$

So the $R_i(i = [1, h] \cap \mathbb{Z})$ is the consistent fuzzy matrix [110]. The score s_d^i of the candidate asset D_{mi} under evaluator U_i can be calculated in Equation (5.10):

$$s_d^i = \frac{\bar{s_d}}{\sum_{l=1}^{k_m} \bar{s_l}}, (\text{where } \bar{s_d} = (\prod_{l=1}^{k_m} r_{el}^i)^{\frac{1}{k_m}} \text{ and } d = [1, k_m] \cap \mathbb{Z}) \tag{5.10}$$

Subsequently, the fuzzy set of weights for each evaluation factor must be calculated, as described by Zou et al. [112], in order to determine the specification efficiency score. This information is derived from the rich relatedness data present in the generated knowledge graph, as well as from the engineer's experience. Two distinct methodologies for asset evaluation are proposed, depending on whether the weights of each evaluation factor are known or not.

The evaluation process involves multiple objectives, which correspond to the evaluation factors. Pareto solution sets will be identified, allowing for the selection of suitable devices accordingly. It is important to note that operation time is not considered in this analysis. In comparison to machining centres, which typically consist of a single workstation, robotic operations involve multiple workstations. As a result, operation time can fluctuate rapidly and exhibit instability. Factors such as matching and changing location do not significantly affect the operation time, as it remains relatively constant during these processes.

In the proposed method, the weight set of each sub-evaluator is defined in A, where A is the weight set of each sub-evaluator in the fuzzy evaluation as shown in Equation (5.11).

$$A = [a_1, a_2, \ldots, a_h] \tag{5.11}$$

Then, the evaluation matrix should be calculated. Many methods can be employed to define the evaluation matrix, such as using the weights sets W_{md}, consistent fuzzy matrix [110], the definition of the membership degree function based on the experts' knowledge [113], historical data, and knowledge graph [114]. The fuzzy matrix, in general, is described in Equation (5.12):

$$R = \begin{pmatrix} r_{11} & r_{12} & \cdots & r_{1g} \\ r_{21} & r_{22} & \cdots & r_{2g} \\ r_{31} & r_{32} & \cdots & r_{3g} \\ \vdots & \vdots & \ddots & \vdots \\ r_{h1} & r_{h2} & \cdots & r_{hg} \end{pmatrix} \tag{5.12}$$

where h is the total number of sub-evaluator sets and g is the total number of evaluation criteria. In terms of utilising a consistent fuzzy matrix to select the most suitable assets from k_m candidate assets under h evaluators, the evaluation criteria can be understood as the

candidate assets, and the fuzzy matrix can then be described as shown in Equation (5.13):

$$R = \begin{pmatrix} S_1^1 & S_2^1 & \cdots & S_{k_m}^1 \\ S_1^2 & S_2^2 & \cdots & S_{k_m}^2 \\ S_1^3 & S_2^3 & \cdots & S_{k_m}^3 \\ \vdots & \vdots & \ddots & \vdots \\ S_1^h & S_2^h & \cdots & S_{k_m}^h \end{pmatrix} \tag{5.13}$$

The total specification efficiency score of each candidate asset S_d can be calculated as in Equation (5.14):

$$S_d = \sum_{k=1}^{h} A_k \cdot s_d^i,$$
$$\text{(where } d = [1, k_m] \cap \mathbb{Z}, \text{and } A \text{ is the evaluator weights set)} \tag{5.14}$$

The ranking of candidate assets can be ascertained by consolidating the influence of h factors, where each candidate asset is assigned a specification efficiency score, S_d, with $d = 1, 2, \ldots, k_m$. The asset possessing the highest specification efficiency score is deemed the most suitable.

To illustrate this concept, consider a hypothetical scenario with three candidate assets (designated as Asset 1, Asset 2, and Asset 3) and two sub-evaluators. The first evaluator is reachability, and the second evaluator is repeatability. The degree to which each candidate asset meets the criteria set by the evaluators is termed the membership degree. This degree can be defined by an engineer based on expertise or derived from a knowledge graph with the support of weight sets W_{md}. The membership degrees for this example are displayed in Table 5.2.

Table 5.2: Membership degrees as defined by the engineer

	Asset 1	Asset 2	Asset 3
Reachability	0.4	0.6	0.7
Repeatability	0.3	0.7	0.2

From the membership degrees, the fuzzy matrix, R, can be constructed as shown in Equation (5.15).

$$R = \begin{bmatrix} 0.4 & 0.6 & 0.7 \\ 0.3 & 0.7 & 0.2 \end{bmatrix}. \tag{5.15}$$

Additionally, assume that the weight sets for each sub-evaluator are as follows:

$$A = [0.3, 0.7]. \tag{5.16}$$

The specification efficiency scores can then be calculated by multiplying matrix A by matrix R, as displayed below:

$$Result = A \cdot R = [0.33, 0.67, 0.35]. \tag{5.17}$$

In this scenario, if only specification efficiency scores are taken into account, Asset 2 would be the most favourable option due to its highest specification efficiency score. This methodology offers a structured process for the selection of candidate assets, enabling optimal decision-making based on predefined sub-evaluators.

5.4.2.2 Cost in Robotic Assembly Cell Reconfiguration

As modern manufacturing is characterised by continually evolving customer demands, reconfiguring robotic assembly cells is indispensable. An evaluation must inform the selection of resources during the reconfiguration of costs. This section describes three costs inherent in robotic assembly cell reconfiguration: installation costs, removal costs, and purchase costs, primarily relating to robots and end effectors. These costs are described below in detail:

- Installation costs, denoted as $Cost_{Installation}$, are the expenses incurred when adding new assets to the existing production line to cater to emerging customer demands.
- Removal costs, represented as $Cost_{Removal}$, are associated with streamlining the production line by excising obsolete or unnecessary assets.
- Purchase costs, represented as $Cost_{Purchase}$, are expenses related to buying new assets, which are assets not currently in the production line but are necessary to meet new customer demands.

The reconfiguration cost, $Cost_{Reconfiguration}$, constitutes a summation of the aforementioned costs and is vital for providing an all-encompassing view of the financial implications of adopting a robotic assembly cell. This cost is mathematically represented in Equation (5.18):

$$Cost_{Reconfiguration} = Cost_{Installation} + Cost_{Purchase} + Cost_{Removal}. \tag{5.18}$$

The process to determine the reconfiguration cost is depicted in Figure 5.5.

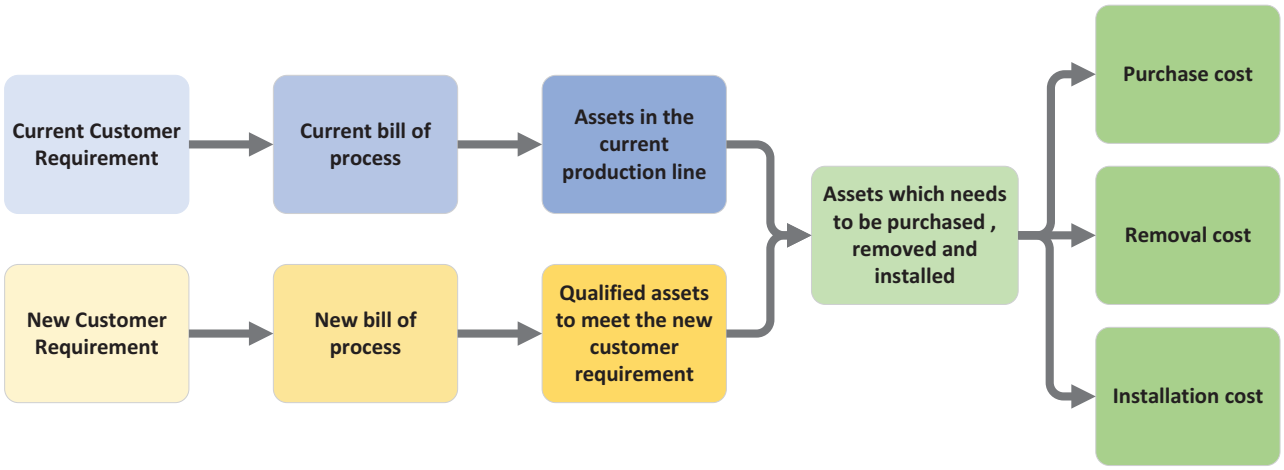


Figure 5.5: Procedure to determine the reconfiguration costs

This figure indicates that to estimate these costs accurately, the current assembly line must be compared with the potential new setup, which includes the new assets required to meet changing customer demands.

5.4.2.3 Additional Evaluators

Further evaluators warrant consideration within the context of asset evaluation in a robotic assembly cell. One such example is an asset's remaining lifespan, which is important in a factory environment due to its potential impact on sustained productivity and maintenance expenditure. An asset approaching the end of its functional life may necessitate increased maintenance or risk unexpected failure, thereby causing disruptions in production. Hence, factoring in the remaining lifespan during the asset selection process aids in making more informed decision-making.

Additionally, an asset's estimated energy consumption could be an evaluator. The energy efficiency of machinery has a considerable effect on operational costs in a manufacturing setting. Energy-efficient machinery can lead to substantial cost savings over time. This methodology focuses on selecting assets before actual operation; therefore, real energy consumption is not considered. Instead, the estimated energy consumption, derived from the manufacturer's specifications or performance data of comparable models, is utilised.

Notably, the selection of evaluators depends on the specific requirements and context of the manufacturing operation. The importance attributed to different evaluators may vary based

on the circumstances.

5.4.3 Evaluator Weights in Asset Evaluation

In the proposed framework, assigning weights to evaluators is fundamental as it delineates the importance of each evaluator within the decision-making mechanism. Two types of evaluators exist: major evaluators and sub-evaluators. Major evaluators primarily concentrate on key elements such as specification efficiency scores and costs, whereas sub-evaluators contribute to the calculation of specification efficiency scores, as explained in Section 5.4.2.1.

On the other hand, weights assigned to major evaluators directly impact the final decision-making process, thus symbolising the priority given to the criteria they evaluate. On the other hand, sub-evaluators indirectly influence decision-making by aiding in the computation of specification efficiency scores, which are then used by the major evaluators.

The careful assignment of weights to both major and sub-evaluators is essential for an accurate and dependable asset evaluation. This procedure must acknowledge the complexity of decision-making within asset evaluation. An effective balance of weights is key for optimal asset selection and the achievement of the intended goals.

Practically, four potential scenarios exist regarding the availability of weights for major and sub-evaluators:

1. Both the weights for major evaluators and sub-evaluators are known.
2. The weights for major evaluators are known, but the weights for sub-evaluators are unknown.
3. The weights for major evaluators are unknown, but the weights for sub-evaluators are known.
4. Neither the weights for major evaluators nor those for sub-evaluators are known.

Each of these scenarios requires different strategies and methodologies to manage the information available effectively and to maintain the reliability of the decision-making process.

Subsequent sections provide methodologies for each scenario to improve the evaluation process.

5.4.3.1 Scenario with Established Weights for Both Major and Sub-Evaluators

In situations where weights for both major and sub-evaluators are already established, the problem reduces to a single-objective problem, simplifying the evaluation process.

For major evaluators, known weights allow for the conversion of a multi-objective optimisation problem into a single-objective problem through the use of scalarisation techniques such as the weighted-sum method. This method simplifies multi-objective optimisation problems by concentrating on a single objective. In the decision-making processes involving asset evaluation for robotic assembly, the method provides a rapid solution that maximises the specification efficiency score and minimises costs. Therefore, the application of established weights is fundamental for strategic decision-making in this context, ultimately enhancing the efficiency and performance of the robotic assembly system. Given the known weights for sub-evaluators, the specification efficiency score can be computed directly using Equation (5.14).

5.4.3.2 Scenario with Known Weights for Major Evaluators and Unknown Weights for Sub-Evaluators

In a situation where weights for major evaluators are known but weights for sub-evaluators are not, this presents some hurdles for decision-making. In this scenario, even though the weights for the major evaluator (including the specification efficiency score) are known, converting this multi-objective problem into a single-objective optimisation problem is complex because the calculation of the specification efficiency score is dependent on knowing the weights of the sub-evaluators.

Without defined weights for the sub-evaluators, calculating an accurate specification efficiency score becomes difficult. Therefore, trying to find an optimal solution in this scenario could lead to ineffective outcomes because of missing information. As a result, this situation is not ideal and can lead to incorrect decisions in the context of asset evaluation in robotic assembly. This underlines the importance of having known weights for both major and sub-evaluators to

ensure an accurate and effective optimisation process.

5.4.3.3 Scenario with Unknown Weights for Major Evaluators and Known Weights for Sub-Evaluators

In contexts where the weights for sub-evaluators are defined but the major evaluator weights remain ambiguous, the challenge of asset selection morphs into a multi-objective optimisation problem. As previously discussed in the literature review, Pareto optimisation is a potent solution to this challenge.

The mathematical embodiment of this optimisation method is encapsulated in Equation (5.19):

$$f_{1,\text{opt}} = \min f_1(x) \qquad f_{2,\text{opt}} = \min f_2(x) ... \qquad f_{n,\text{opt}} = \min f_n(x) \tag{5.19}$$

where $f_i(x)$ represents each distinct objective function, with the aim being to discern the minimal solution for every function. The Pareto method ensures that the elements within solution vectors maintain their independence during optimisation, enabling a balanced evaluation of each objective.

Figure 5.6 portrays the intricacies of multi-objective optimisation. The visualisation demonstrates the simultaneous consideration of myriad objectives, each with its own distinct weight, collectively contributing to a holistic evaluation of prospective assets.

By employing the Pareto optimisation method in contexts with undefined major evaluator weights, a balanced evaluation of multiple objectives becomes feasible. This facilitates the identification of optimal solutions and enhances the overall performance of robotic assembly systems.

5.4.3.4 Scenario with Unknown Weights for Major and Sub-Evaluators

When both the major and sub-evaluator weights are unknown, the problem becomes more challenging than in the other scenarios. The optimisation problem turns into a multi-objective one because there are no defined weights for the major evaluators. With these unknowns, the

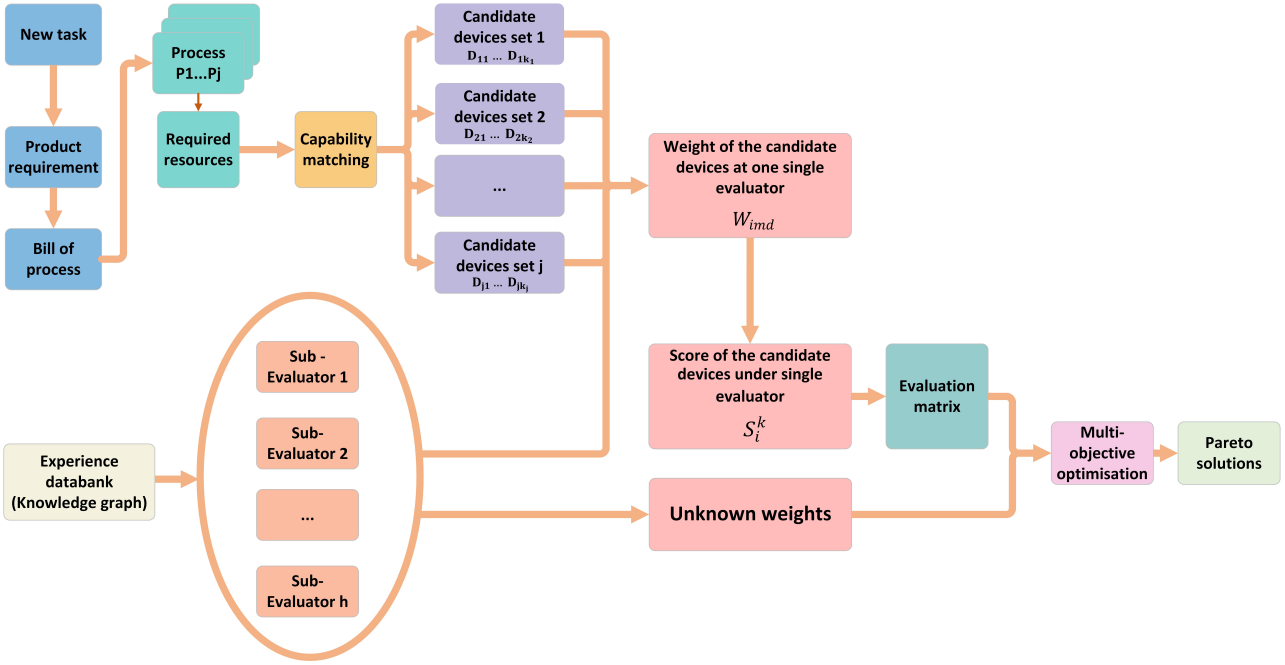


Figure 5.6: Asset selection in the scenario of unknown weights of major evaluators

objectives for optimisation change, as shown in Table 5.3.

Table 5.3: Changed optimisation objectives and targets

Changed Optimisation Objectives	Target
specification efficiency score for sub-evaluator 1	Maximise
specification efficiency score for sub-evaluator 2	Maximise
specification efficiency score for sub-evaluator ...	Maximise
specification efficiency score for sub-evaluator n	Maximise
reconfiguration cost	Minimise

On the one hand, the specification efficiency score for each sub-evaluator becomes a separate objective that should be maximised. On the other hand, the total cost, which is the sum of the purchase cost and reconfiguration cost, is an objective that should be minimised.

To manage the complexity that arises from not knowing the weights of the sub-evaluators, a solution could be to calculate the specification efficiency score for each sub-evaluator on its own. These separate specification efficiency scores can then be treated as different objectives in the decision-making process.

Although this approach is complex, it yields a possible solution when there are no predefined weights. However, the results of this method might not be as clear or efficient as when the weights are known beforehand.

In conclusion, when both the major and sub-evaluator weights are unknown, creative problem-

solving is required, along with a thorough understanding of the complex nature of decision-making in asset evaluation for robotic assembly. Even though this scenario is more complex than the previously described scenarios, it provides an opportunity to trial different methods for managing the complexity of multi-objective optimisation.

5.5 Chapter Summary

This chapter addresses RQ2 by exploring asset choices in RMSs, focusing on robotic assembly cells. The approach involved a holistic view, considering a range of factors such as cost, energy use, the remaining life of assets, changes over time, and resource use. These factors are key in determining the overall efficiency of the manufacturing process. A crucial point was the changing and evolving nature of these factors and their significant role in the asset selection process. Since this problem has many aspects, the process must be continually updated and revised based on new information and changing manufacturing needs. To manage the complexity and details of this process, this chapter suggests strategic approaches such as multi-objective optimisation techniques. These strategies are designed to deal with a combination of discrete and continuous decision variables effectively, thereby improving the efficiency and accuracy of the asset selection process. In summary, this chapter explains the asset selection process within the context of reconfigurable manufacturing systems. The suggested approach offers manufacturers the necessary tools and knowledge to optimise the asset selection process, allowing for better flexibility in response to changing needs in today's manufacturing environment.

Chapter 6

Layout Configuration Optimisation

This chapter aims to address RQ3: "How can a reconfigurable robotic assembly system efficiently optimise its operations, especially in layout, after adapting to current process requirements?" and Objective 3. The primary emphasis is on the development of a modular framework for enhancing the layout of the robotic assembly cell, which outlines a multi-objective modular layout optimisation framework designed within a simulation environment.

6.1 Introduction

As introduced in Chapter 5, the initial reconfiguration stage in my proposed methodology involves selecting essential assets for the robotic assembly process. While this sets the foundation, the subsequent phase—focusing on the intricate details of layout configurations, refinement of process parameters, and task sequences for each robot—remains a complex challenge. In this chapter, this critical phase is explored, with an emphasis on layout optimisation, a key factor in enhancing the efficiency and effectiveness of the assembly process.

To address RQ3, this chapter introduces a modular layout optimisation framework that operates within a simulation environment, enabling multi-criteria optimisation. This framework is not just theoretical but grounded in practical implementation details. It systematically navigates through the process of optimising the layout of robotic assembly cells, ensuring that both individual machine and system-level considerations are integrated. The framework's adaptability allows it to be utilised with various simulation software, making it a versatile tool for

different manufacturing contexts.

Central to this framework is the clear and detailed outline of its implementation. This begins with the identification of decision variables, such as the coordinates and rotation angles of assets within the manufacturing system. These variables are the levers through which the layout can be manipulated and optimised. Following this, the framework establishes specific objectives, like minimising total cycle time or maximising overall throughput, and sets constraints to ensure practical and feasible solutions. The core of the optimisation process is then carried out through a carefully selected set of algorithms, each chosen for their suitability to the specific layout challenges at hand.

This chapter describes each step of the layout optimisation process in detail. It explains how the simulation environment is used to evaluate potential layouts, how the optimisation environment processes and analyses this data, and how the chosen algorithms navigate towards the most efficient layout configuration. By the end of this chapter, the reader will have a comprehensive understanding of not just the theoretical underpinnings of the modular layout optimisation framework but also its practical implementation, ensuring that the reconfigurable robotic assembly system can optimally adapt to varying process requirements.

6.2 Optimisation Framework for Robotic Assembly Cell

In the realm of robotic assembly manufacturing, the development of a coherent and effective optimisation framework is of paramount importance. Three interconnected components are central to this framework: the simulation environment, the optimisation environment, and the optimisation algorithms.

First, the simulation environment is a pivotal tool that emulates real-time assembly processes with high fidelity. This emulation allows for the precise evaluation of potential reconfigurations without any interference in ongoing manufacturing operations, making it essential during the intricate second reconfiguration phase.

Second, the optimisation environment serves as the nexus of this framework. It is designed to be adaptive, with objectives based on specific criteria such as cycle time, energy consumption, and cost. Yet, while it ambitiously pursues these objectives, it remains grounded by certain

constraints, notably budgetary ones. It is a software and programming environment to do the optimisation tasks.

Lastly, the modular optimisation algorithms form the strategic core of the framework. Stored diligently within the experience databank, they explore a multitude of configurations, seeking the most optimal fit for a specific product in the assembly line. The results, which emerge from the most effective algorithm, lay the foundation for the successful reconfiguration of the robotic assembly cell.

Figure 6.1 elucidates the symbiotic relationship between the three primary components and how they coalesce to form the larger framework.

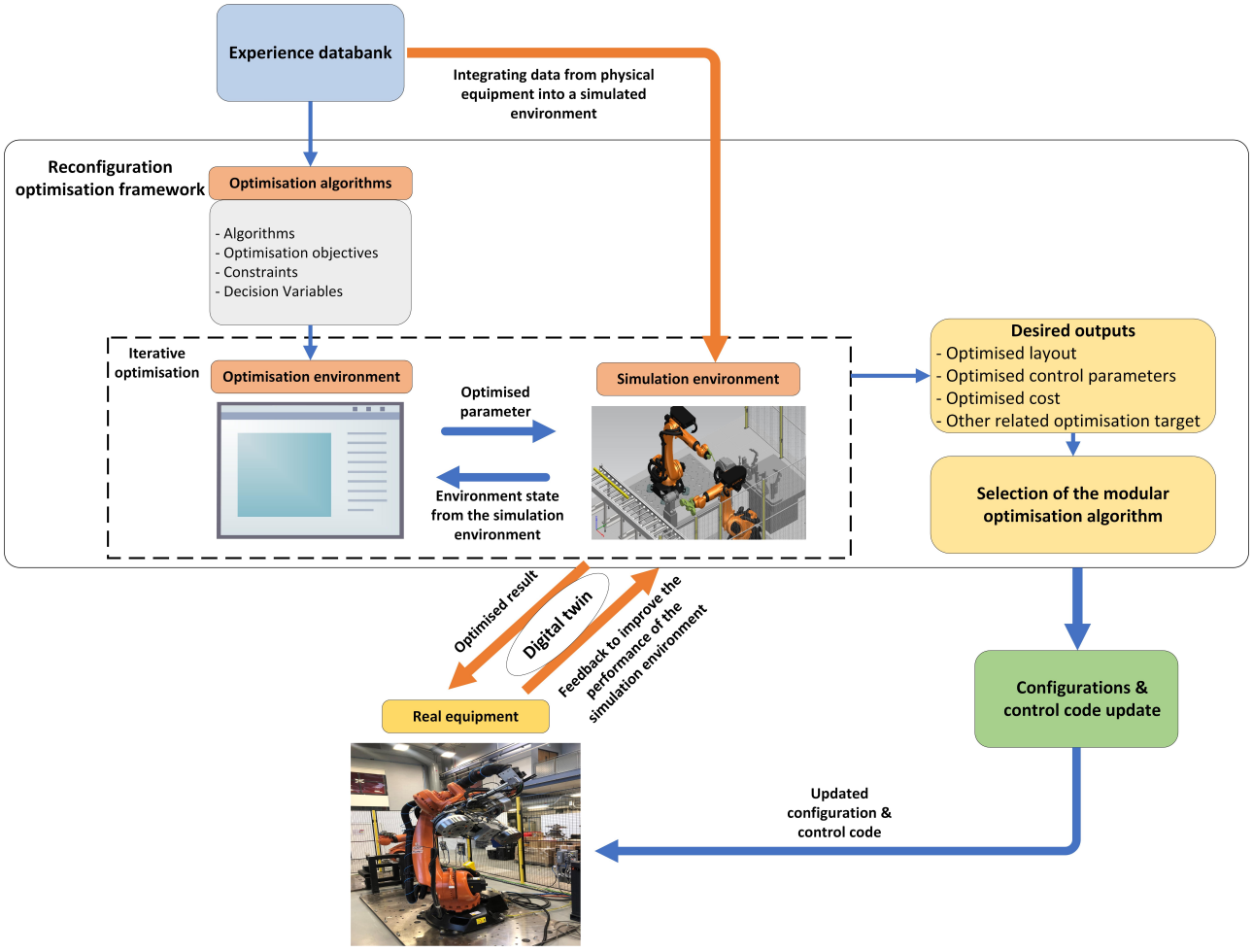


Figure 6.1: Details of utilising the reconfiguration optimisation framework

6.2.1 Simulation Environment

In the realm of robotic assembly cells within reconfigurable manufacturing systems, the simulation environment emerges as the foundation for grasping and fine-tuning intricate processes. Such environments act as controlled virtual canvases where scenarios unfold, hypotheses are rigorously tested, and experiments come to life. An in-depth investigation into the inherent merits of simulation environments for optimisation reveals several key attributes.

First, these environments champion flexibility. They readily embrace a diverse spectrum of scenarios and system configurations. While mathematical models might demand significant revamps when adapting to system modifications, simulation environments often handle these transitions with more fluidity. Second, these environments prioritise realism. Even though specific tools, such as Tecnomatix Process Simulate [70], might occasionally grapple with portraying stochastic elements or non-linear dynamics, the overarching objective of simulation environments remains to mirror real-world intricacies.

Beyond flexibility and realism, simulation environments are designed for iterative improvement. They serve as platforms where strategies undergo repeated testing and fine-tuning to unearth potential lapses or choke points that might elude purely theoretical frameworks. Lastly, integration with actual data sets these environments apart. They can resonate with real-world data streams, churning out feedback and validation in real time. This ensures that the ensuing results are not merely theoretical constructs but are grounded, relevant, and ripe for implementation.

6.2.2 Optimisation Environment

A dynamic optimisation environment is decisive for an effective robotic assembly cell framework. The environment, capable of processing and analysing data generated by simulation setups, can be flexibly deployed both locally and on cloud platforms, contingent on the specific needs and constraints of the given scenario.

Effective communication between the simulation and optimisation environments is paramount. Locally, protocols such as Message Queuing Telemetry Transport (MQTT), shared memory, socket communication, and OPC Unified Architecture (OPC UA) serve the purpose, with

OPC UA being particularly noteworthy for its industrial ubiquity. In cloud deployments, technologies such as REST APIs, WebSockets, and services such as Amazon SQS and Azure Service Bus ensure efficient data exchange, supporting both voluminous data transfers and real-time communication.

6.2.3 Optimisation Algorithms

The optimisation environment for robotic assembly cell reconfiguration is a critical component in data-driven decision-making frameworks, enabling the efficient processing and analysis of data generated by simulation environments. The selection of the optimal algorithm is a critical component of the optimisation environment, and different categories of optimisation algorithms can be utilised to optimise the simulation environment and achieve the desired optimisation targets.

6.2.3.1 Algorithm Types

Optimisation algorithms can be broadly categorised into several types based on their underlying principles and characteristics. Exact algorithms guarantee an optimal solution but may be computationally expensive for large problems. Heuristic algorithms provide fast and efficient solutions but may not guarantee an optimal solution. Meta-heuristic algorithms constitute a type of heuristic algorithm that can be applied to a wide range of problems and often utilise search strategies inspired by natural phenomena or social behaviour. Meta-heuristic algorithms can be further categorised into evolutionary algorithms, swarm intelligence algorithms, and other meta-heuristics, such as simulated annealing and tabu search. Evolutionary algorithms use genetic operations such as mutation and crossover to evolve a population of potential solutions over multiple generations. Swarm intelligence algorithms are inspired by the collective behaviour of social animals and use a decentralised approach to find optimal solutions. Other meta-heuristics, such as simulated annealing and tabu search, are based on probabilistic and memory-based search techniques.

6.2.3.2 Algorithm Application in the Robotic Assembly Cell

In the context of optimisation environments for robotic assembly cell reconfiguration, the selection of the appropriate algorithm depends on the specific problem and optimisation target. For example, exact algorithms may be suitable for problems with small search spaces, while heuristic algorithms may be more appropriate for large-scale optimisation problems. Meta-heuristic algorithms can be applied to a wide range of problems and have been shown to be effective in the context of robotic assembly cell reconfiguration, such as optimising layout designs, scheduling, and routing of materials and parts in the assembly line. Evolutionary algorithms and swarm intelligence algorithms can be used to optimise the parameters of the robotic assembly cell, such as robot trajectories, movements, and configurations. Machine learning algorithms, such as deep learning and reinforcement learning, can be used for highly complex and difficult-to-represent problems, such as optimising robot movements based on the assembly line layout and task requirements.

6.2.3.3 Criteria for Selecting the Algorithm

Selecting the best optimisation algorithm for a specific problem is a critical component of designing an optimisation environment that effectively meets specific requirements and leads to improved efficiency and productivity. To identify the most suitable optimisation algorithm, several criteria must be considered, including solution quality, efficiency, robustness, scalability, adaptability, transparency, and simplicity.

Solution quality is a crucial criterion, as the algorithm should be able to provide high-quality solutions that are as close to the optimal solution as possible. Efficiency is also an important criterion, as the algorithm should be able to find a solution within a reasonable time frame, especially for large-scale optimisation problems. Additionally, robustness is key, as the algorithm should be able to handle noisy, incomplete, or uncertain data and still provide suitable solutions.

Simplicity is also essential, as the algorithm should be simple to implement and use, thereby reducing the overall complexity of the optimisation environment. By considering these criteria, researchers and practitioners can identify the most suitable optimisation algorithm for their specific problem and optimise the simulation environment accordingly.

The choice of the appropriate algorithm and parameters can significantly impact the performance and effectiveness of the optimisation environment. Therefore, selecting the best algorithm for the problem at hand is essential for achieving optimal performance and ensuring successful implementation of the optimisation framework. The experience databank stores information on the desired outputs and the results of the selection of the modular optimisation algorithm, enabling the optimisation environment to learn and improve over time. In summary, the selection of the appropriate optimisation algorithm is a critical component of the optimisation environment for robotic assembly cell reconfiguration, and the various categories of optimisation algorithms offer different approaches to address specific optimisation problems.

6.2.3.4 Decision Variables for Layout Optimisation

When it comes to layout optimisation, the decision variables in the proposed framework of robotic assembly cells play a pivotal role in determining the optimal arrangement of manufacturing cells, workstations, and equipment. This framework, therefore, adopts a comprehensive approach to these variables, ensuring that the layout caters to the specific needs of the robotic assembly cell environment. The position-related information within these decision variables can be categorised into three main types: coordinates, orientation, and transformation matrix. Each of these categories is detailed below.

6.2.3.4.1 Coordinates The precise placement of robots, workstations, and other equipment within the assembly cell is essential for production efficiency and material flow. The spatial arrangement of equipment in the cell can be intuitively described using a 3D Cartesian coordinate system, represented as (x, y, z).

6.2.3.4.2 Orientation The orientation of equipment within the assembly cell dictates how the equipment interacts with the environment and other entities. Several methods exist to represent this orientation:

1. Roll, Pitch, and Yaw: These Euler angles describe the orientation of an object in 3D space. Denoted as ϕ (roll), θ (pitch), and ψ (yaw), these angles correspond to rotations

about the principal axes of a body-fixed coordinate system:

$$\begin{aligned}\phi &= \arctan\frac{y}{z},\\ \theta &= \arctan\frac{x}{\sqrt{y^2+z^2}},\\ \psi &= \arctan\frac{\sqrt{x^2+y^2}}{z}.\end{aligned} \tag{6.1}$$

2. Quaternions: Representing orientation, quaternions are a four-dimensional vector $q = (q_0, q_1, q_2, q_3)$ that denotes rotation about an arbitrary axis. The relationship between the Euler angles and quaternions can be expressed as:

$$\begin{aligned}q_0 &= \cos\frac{\phi}{2}\cos\frac{\theta}{2}\cos\frac{\psi}{2} + \sin\frac{\phi}{2}\sin\frac{\theta}{2}\sin\frac{\psi}{2},\\ q_1 &= \sin\frac{\phi}{2}\cos\frac{\theta}{2}\cos\frac{\psi}{2} - \cos\frac{\phi}{2}\sin\frac{\theta}{2}\sin\frac{\psi}{2},\\ q_2 &= \cos\frac{\phi}{2}\sin\frac{\theta}{2}\cos\frac{\psi}{2} + \sin\frac{\phi}{2}\cos\frac{\theta}{2}\sin\frac{\psi}{2},\\ q_3 &= \cos\frac{\phi}{2}\cos\frac{\theta}{2}\sin\frac{\psi}{2} - \sin\frac{\phi}{2}\sin\frac{\theta}{2}\cos\frac{\psi}{2}.\end{aligned} \tag{6.2}$$

6.2.3.4.3 Transformation Matrix The transformation matrix is a mathematical representation that combines both position and orientation information, allowing for the representation of linear transformations, including rotations, translations, and scaling. This matrix provides a comprehensive description of an object's position and orientation in the assembly cell. The use of this matrix in the optimisation framework leads to a holistic approach to layout optimisation, encompassing both spatial and orientational considerations.

By incorporating these three categories of position information into the decision variables, the optimisation framework can generate layouts that are efficient and adapt to the intricate requirements of robotic assembly cells.

6.2.3.5 Constraints

In layout optimisation for a robotic assembly cell, the constraints that govern feasible solutions must be identified and categorised. As mentioned in Section 4.2.2.9, these constraints can be broadly classified into equality and inequality constraints. Equality constraints impose strict conditions on the layout configuration that must be satisfied beforehand, while inequality con-

straints set upper or lower bounds on the variables, ensuring that the optimal layout operates within these bounds.

A solution can be developed by taking into account these equality and inequality constraints during the layout optimisation process. This ensures that the robotic assembly cell operates efficiently and safely while adhering to the specific requirements of the manufacturing environment. Optimisation objectives, such as minimising cycle time, can then be pursued within the boundaries established by these constraints.

6.2.3.6 Optimisation Objectives

To effectively optimise the layout of a robotic assembly cell, it's essential to consider a blend of various optimisation objectives, as each plays a pivotal role in enhancing the overall performance and efficiency of the manufacturing process. As mentioned in Section 4.2.2.9, layout optimisation has several optimisation objectives. This section discusses several optimisation objectives, including cycle time, robot manoeuvrability, and energy consumption. The emphasis on specific objectives can vary based on customer requirements and the unique operational goals of the manufacturing setup. In some cases, a single objective may be the focus, while in others, a multi-objective approach is required to meet broader operational criteria.

6.2.3.6.1 Cycle Time Cycle time, a critical performance metric in robotic assembly cells, is the total time a robot needs to complete one operational cycle. It consists of three primary components: operation time, movement time, and waiting time. Lower cycle times correspond to higher efficiency. The total cycle time is expressed as shown in Equation (6.3):

$$t_{cycle} = t_{operation} + t_{movement} + t_{wait} \tag{6.3}$$

Optimising operation cycles is vital for enhancing efficiency and productivity in the manufacturing process. For layout optimisation, two principal methodologies facilitate cycle time calculation:

1. **Mathematical Modelling:** This approach utilises analytical methods, including inverse kinematics and movement equations, to calculate cycle time, and it involves various steps:

- The operation time refers to the duration a robot takes to actively perform its designated task, such as picking up an object, assembling a component, welding, or painting. It contrasts with the time spent moving between tasks or waiting. Typically, operation time is a predefined or known quantity.
- The movement time correlates with the distance between the layout elements of the robotic cell. It can be optimised using layout optimisation techniques. With inverse kinematics, joint angles at the start and endpoints of a robot's motion can be calculated, and with the maximal angular velocities of the robot's joints, the movement time can be computed.
- The waiting time is influenced by the robot's arrangement. It can be minimised using effective scheduling and resource allocation strategies.

If the maximum angular velocity of the robot R is known, denoted as ω_1, ω_2 ,..., ω_n, (where n is the number of joints), and the coordinates of the robot's start and end points, the inverse kinematics at the endpoint coordinates can be solved to obtain f feasible sets of joint angles.

The joint angle at the starting point is given by Equation (6.4):

$$\theta = [\theta_1, \theta_2..., \theta_n] \tag{6.4}$$

The joint angles at the endpoint are given by Equation (6.5):

$$\theta = [{}^i\theta_1, {}^i\theta_2... {}^i\theta_n], \text{where i} = 1, 2, ..., \text{f} \tag{6.5}$$

Due to the existence of multiple inverse solutions and corresponding joint angles, it is essential to select the solution that results in the minimum time. This process entails calculating the motion cost for each inverse solution using Equation (6.6):

$$Value_i = \sum_{j=1}^{n} \frac{|\theta_j - {}^i\theta_j|}{\omega_j} \tag{6.6}$$

Once the motion costs for all solutions are determined, the smallest cost is identified using Equation (6.7):

$$Value_{min} = \min_{i=1,...,n} (Value_i) \tag{6.7}$$

The solution corresponding to $Value_{min}$ represents the most efficient movement, and it is the one that should be selected for optimal robot operation.

Assuming k is the appropriate solution with the minimum joint motion cost, the movement time can be calculated as in Equation (6.8):

$$t_{movement} = \max_{j=1,\ldots,n} \left(\frac{|\theta_j - {}^{k}\theta_j|}{\omega_j} \right) \tag{6.8}$$

2. **Simulation Software or Real-Time Data Recording:** This method uses sophisticated simulation software or real-time data recording to obtain precise cycle time data. Digital twin technology is particularly useful, as it provides accurate cycle time information directly after the operational points and paths have been determined in the simulated environment.

6.2.3.6.2 Robot's Manoeuvrability The manoeuvrability of a robot pertains to the capability of the robot's end effector to operate at a specific point in space, demonstrating the dynamic reversibility of the robot. This ability is critical for a robot's end effector when performing tasks that require a diverse set of complex actions within a substantial spatial range. This flexibility can, however, be compromised when the robot is at a singularity, losing one or more degrees of freedom.

Wrist singularity, elbow singularity, and shoulder singularity are three specific types of singularity that could occur, which are distinguished based on the robot's configuration and type. Wrist singularity occurs when the robot's wrist axes intersect at a single point, resulting in a loss of one degree of freedom and locking the robot's wrist. Elbow singularity occurs when the robot's arm is either fully extended or fully retracted, thus causing the robot's end effector to lose the ability to move in certain directions. Shoulder singularity occurs when the robot's shoulder joint is in line with the base joint, causing the robot to lose one degree of freedom and restricting the movement of the arm in certain directions.

For a six-axis industrial robot, for example, the wrist, elbow, and shoulder singularities are depicted below.

1. Wrist singularity is a specific type of singularity that can occur in six-axis industrial robots. It refers to a configuration where the axes of the last three joints of the robot arm,

often referred to as the wrist joints, become aligned or parallel. This alignment causes the robot to lose a degree of freedom, leading to limitations in its motion capabilities. When a six-axis robot enters a wrist singularity, precisely controlling the orientation of the robot's end effector becomes challenging. Although the robot can still move and perform tasks, it may experience difficulties maintaining the desired orientation of the end effector.

2. Elbow singularity occurs explicitly when the robot's second and third axes (usually the shoulder and elbow joints) align with each other in a straight line. This configuration results in a loss of degree of freedom, as the robot has no unique direction in which to extend or contract its arm to reach a target point. In this case, the arm can still pivot around the base joint or rotate around its own axis, but it cannot reach out to different points in its workspace as effectively as it usually would.

3. Shoulder singularity occurs when the robot's first axis (usually the rotation of the base or waist of the robot) and fourth axis (the first axis of the wrist) become aligned or parallel – in other words, when the wrist centre aligns with the axis of the base rotation. This results in a loss of a degree of freedom: the robot cannot move freely in all directions and consequently has difficulty maintaining the orientation of the end effector while moving its position.

The Jacobian matrix can be a descriptor of a robot's manoeuvrability. Specifically, the Jacobian determinant's value at a given point in space reflects the robot's ability to manoeuvre around that point, indicating the robot's flexibility. For a robot with k joints, the joint vector can be represented as in Equation (6.9):

$$\theta = [\theta_1, \theta_2, \theta_3, ..., \theta_n]^T \tag{6.9}$$

The end effector pose of the robot can be denoted as:

$$X = [x_1, x_2, x_3, ..., x_m]^T, \text{(where } m \text{ represents the degree of freedom of the robot)} \tag{6.10}$$

The end effector pose is a function of the joint vector, as follows:

$$X = f(\theta) = \begin{pmatrix} f_1(\boldsymbol{\theta}) \\ f_2(\boldsymbol{\theta}) \\ \vdots \\ f_m(\boldsymbol{\theta}) \end{pmatrix} \tag{6.11}$$

Deriving both sides of the equation results in

$$\dot{\boldsymbol{X}} = \boldsymbol{J}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} \tag{6.12}$$

where $J(\theta)$ is the Jacobian matrix, $\dot{X}$ is the Cartesian velocity vector, and $\dot{\theta}$ is the joint angular velocity vector. The Jacobian matrix of the function f is an $m \times n$ matrix, defined as:

$$\boldsymbol{J}(\boldsymbol{\theta}) = \begin{pmatrix} \frac{\partial f_1}{\partial \theta_1} & \frac{\partial f_1}{\partial \theta_2} & \cdots & \frac{\partial f_1}{\partial \theta_n} \\ \frac{\partial f_2}{\partial \theta_1} & \frac{\partial f_2}{\partial \theta_2} & \cdots & \frac{\partial f_2}{\partial \theta_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial \theta_1} & \frac{\partial f_m}{\partial \theta_2} & \cdots & \frac{\partial f_m}{\partial \theta_n} \end{pmatrix} \tag{6.13}$$

For any group of joint angles of the robot θ^*, the Jacobian matrix is not of full rank when it satisfies the following condition:

$$\operatorname{rank} J(\theta^*) < m \tag{6.14}$$

Under this condition, the robot is in a singularity position, and the robot loses some degrees of freedom, which should be avoided.

The determinant of the Jacobian matrix serves as an indicator for detecting singularities in the robot. Specifically, when the determinant equals zero, the Jacobian matrix is singular (not full rank), indicating that the robot is in a singularity. Conversely, when the determinant approaches zero, it suggests that the robot is nearing a singularity. The value of the determinant can be calculated using Equation (6.15):

$$w = |\det J(\theta)| \tag{6.15}$$

The total robot manoeuvrability index W_{total} of the entire robotic assembly unit is the sum of the corresponding Jacobian matrix determinants of all the working points of the equipment,

expressed as:

$$W_{total} = \sum_{l}^{R} \sum_{j}^{op^l} w_{l,j} \tag{6.16}$$

where $w_{l,j}$ represents the manoeuvrability index of robot l at the operation point j, R is the total number of robots in the production cell, and op^l represents the number of operations that robot l must execute.

The following texts provide examples of the manoeuvrability index of a robot. First, the workspace of the robot, as depicted in Figure 6.2, should be defined.

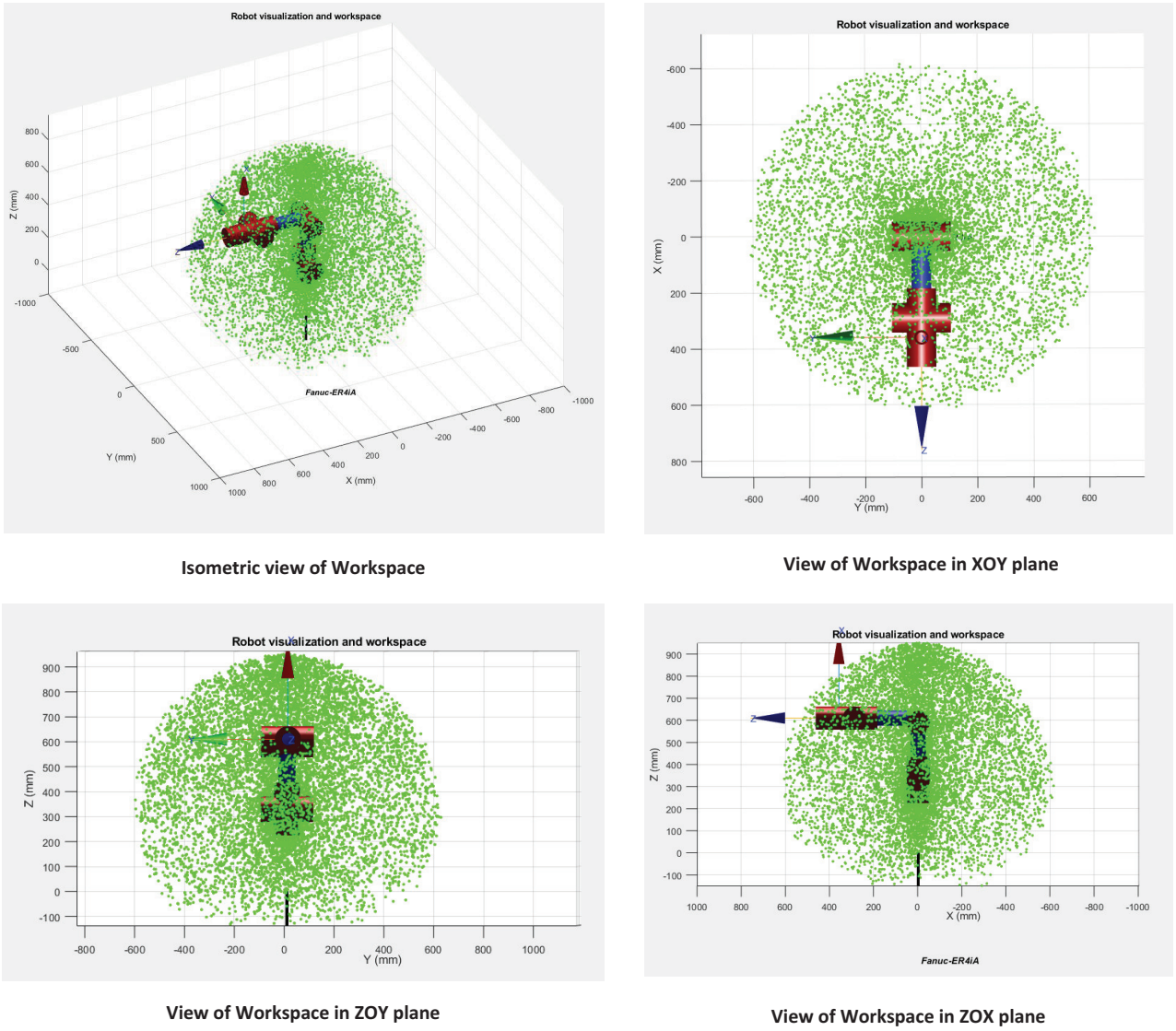


Figure 6.2: Workspace of the FANUC ER-4iA robot

In the context of a robotic workspace, certain positions can be conducive to the execution of operations, specifically when the determinant of the Jacobian matrix is sufficiently large.

Conversely, other positions may prove less effective for operations, particularly when the denominator of the Jacobian matrix approaches zero, or the matrix is not of full rank. To visually represent this information, the following procedure is followed:

1. **Computation of the Absolute Values of the Determinants**

   Initially, the robot is assigned 10,000 sets of joint values. The determinant of the Jacobian matrix can be computed using Equations (6.9), (6.10), (6.11), (6.12), (6.13), and (6.15). The resulting information is stored within the array denoted as J. Subsequently, the absolute values of these determinants (in other words, robot manoeuvrability index), are computed and housed in a new array, J_{abs}. The equation for the i-th element is formulated as follows in Equation (6.17):

   $$J_{abs}[i] = |J[i]| \tag{6.17}$$

2. **Identification of the Minimum and Maximum Values of the Absolute Determinants**

   After the construction of the J_{abs} array, the minimum and maximum values contained within this array are identified, represented as $J_{abs_{min}}$ and $J_{abs_{max}}$, respectively. Mathematically, these can be expressed using Equations (6.18) and (6.19):

   $$J_{abs_{min}} = \min(J_{abs}) \tag{6.18}$$

   $$J_{abs_{max}} = \max(J_{abs}) \tag{6.19}$$

3. **Scaling of the Absolute Determinants**

   At the analysis' concluding phase, the absolute determinants undergo scaling. This process, which normalises the determinants, involves dividing them by an appropriately chosen scaling factor, thereby ensuring comparability across the dataset:

   $$J_{scaled}[i] = \frac{J_{abs}[i]}{scaling\_factor} \tag{6.20}$$

   Referencing Equation (6.20), the scaling factor is crucial and must be selected to suit the dataset's range, preserving the comparability of the results. As an illustrative example, when the scaling factor is the range between the maximum and minimum determinants, the scaling operation is described by:

$$J_{scaled}[i] = \frac{J_{abs}[i]}{J_{abs_{max}} - J_{abs_{min}}} \tag{6.21}$$

Equation (6.21) demonstrates this specific instance of scaling. However, the primary intent of scaling, as indicated by Equation (6.20), is to facilitate the visual comparison of the determinants, without altering the inherent relationships among them. It is imperative that the selected scaling factor maintains the original proportionate relationships and ensures that the scaled values are meaningful and clearly comparable.

Upon completion, $J_{scaled}[i]$ can be viewed as w, as denoted in Equation (6.15). The $J_{scaled}[i]$ is called the robot manoeuvrability index. To visually represent this information, an updated workspace diagram of the robot is created. This diagram employs a variety of colours to represent the points based on the value of J_{scaled}. This process enhances the robot's workspace visualisation, as it provides insights into the robot's manoeuvrability, as depicted in Figure 6.3.

- $J_{scaled} < 0.2$, red
- $0.2 \leq J_{scaled} < 0.4$, orange
- $0.4 \leq J_{scaled} < 0.6$, yellow
- $0.6 \leq J_{scaled} < 0.8$, green
- $0.8 \leq J_{scaled}$, blue

In the context of robot manipulability, the Jacobian matrix determinant is used as a measure of the manipulability index, indicating the robot's ability to move in different directions without encountering a singularity. A larger Jacobian matrix determinant, in absolute terms, implies superior manipulability, as the robot is able to move more freely in various directions than robots with a smaller determinant without encountering a singularity. However, a large determinant does not necessarily mean improved manipulability, as the determinant can be affected by the units used to measure the robot's joints and end-effector position. Conversely, suppose the Jacobian matrix determinant equals zero (or near zero). In that case, it indicates that the robot is in a singular position, where the Jacobian loses rank, and the robot loses certain degrees of freedom. Thus, the robot's configuration and joint angles must be accounted for, with a focus on situations where the determinant nears zero, indicating potential singularity, rather than solely on the determinant's absolute size.

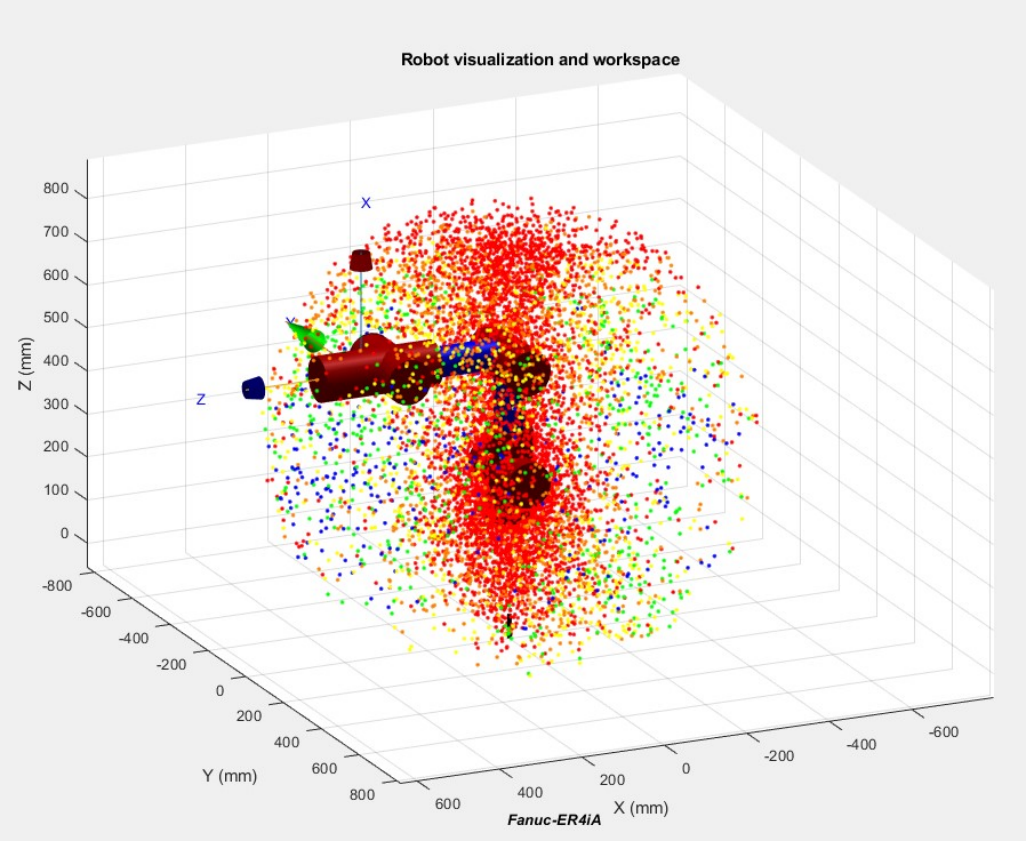


Figure 6.3: Updated workspace of the FANUC ER-4iA robot

6.2.3.6.3 Energy Consumption Energy consumption is a central element in the optimisation of layout configurations for robotic assembly cells within RMSs, targeting both sustainability and operational efficiency. Specifically, two primary methodologies enable the measurement of energy consumption, offering pivotal data for more strategic optimisation decisions:

1. **Energy Consumption Estimation:** This methodology utilises known parameters, such as the velocity and acceleration of the robots in the system, to estimate the energy consumption associated with different layout configurations.

2. **Real-Time or Simulated Environment Energy Consumption Measurement:** This alternate methodology facilitates the capture of energy consumption data either in real time or within a simulated environment. This experiential data yields vital insights into energy efficiency under actual operational circumstances.

In conclusion, an extensive understanding of energy consumption underpins the optimisation process of layout configurations in robotic assembly cells within RMS. The collection of accurate energy consumption data, whether via simulation or real-time recording, coupled with

a comprehensive optimisation strategy, can aid in developing energy-efficient layout configurations. The choice between these methodologies is contingent upon the specific real-world application.

6.3 Chapter Summary

This chapter explored the development and application of a unique modular optimisation framework tailored to support the design of reconfigurable manufacturing systems to address RQ3, emphasising robotic assembly cells. The core strength of this framework stems from its intricate and systematic structure, which encompasses three primary components: the simulation environment, the optimisation environment, and the optimisation algorithms.

A key highlight of this modular framework is the application of layout optimisation in robotic assembly cells. Addressing crucial equality and inequality constraints that affect the feasible solutions can ensure that the optimised layout operates efficiently and safely within the specific requirements of the manufacturing environment.

Incorporating the experience databank into this process yields a significant competitive edge. This knowledge graph streamlines the decision-making process, offering valuable recommendations based on the gathered information. The integration of this tool gives this research an advantage over traditional methods, showcasing the potential of knowledge-aided systems in modern manufacturing design.

Chapter 7

Software Implementation

This chapter moves from theoretical concepts to their practical application, focusing on software development for the proposed methodology as described in Chapter 4, 5 and 6. It begins the process of fulfil Objective 4, which will continue and be completed in Chapter 8.

7.1 Introduction

In the discussion on the experience databank, a central repository emerged, designed to compile extensive knowledge that encompasses tasks, processes, assets, and reconfiguration intricacies. Section 7.2 elaborates on the software interface conceptualised for this databank, utilising ontology models and knowledge graph methodologies.

Section 7.3 then discusses the asset selection methodology, highlighting its software embodiment. Beyond its theoretical underpinning, asset selection in practice involves a blend of algorithms and decision matrices. A dedicated software tool employing knowledge graphs and multi-criteria decision-making algorithms is thus developed to facilitate this intricate process.

This chapter wraps up with a discussion on the software approach to layout optimisation, detailed in Section 7.4. Here, a synergy of knowledge graphs, simulation environments, and modular artificial intelligence tools becomes evident. Integration with platforms such as Tecnomatix Process Simulate and RoboDK ensures that the simulations align with prevailing industry standards.

In summary, this chapter connects the theoretical constructs of earlier sections to their tangible, real-world applications and paves the way for subsequent discussions on real industry use cases.

7.2 Interface Development for Experience Databank

As discussed in Chapter 4, the knowledge graph (experience databank) comprises both the schema layer and the entity layer. This section details the development of interfaces for these layers. As depicted in Figure 7.1, the procedure encompasses creating and connecting to the graph database, uploading the ontology model (which corresponds to the schema layer described in Chapter 4), loading the dataset, and subsequently generating the knowledge graph (representing the entity layer as outlined in Chapter 4).

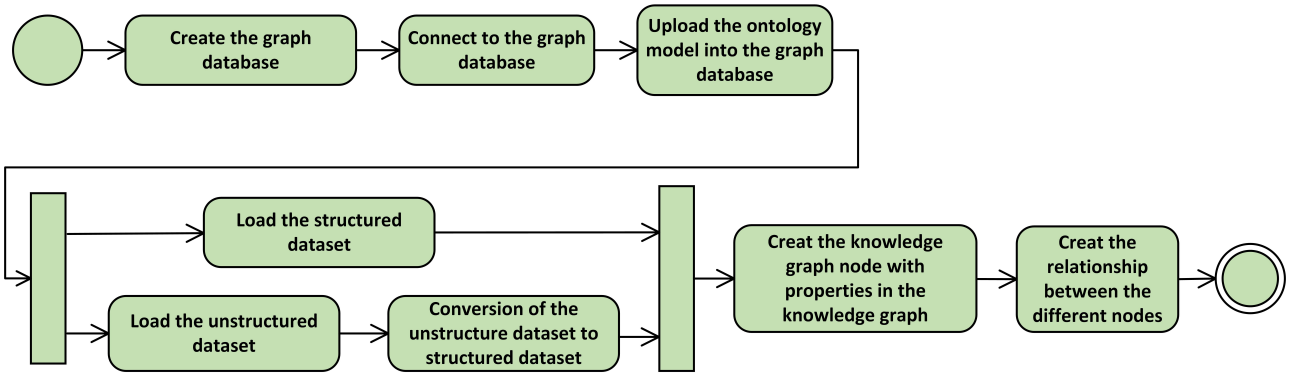


Figure 7.1: UML activity diagram for building the experience databank

7.2.1 Software selection

As described in Section 2.6, among the myriad tools available for designing and refining the schema layer of a knowledge graph, Protégé was chosen, predominantly for its efficiency, flexibility, scalability, ease of use, and integration capacity, as well as the following reasons:

- **Maturity and Adoption**: Protégé has been in existence for decades and is one of the most widely used ontology editors in both academic and industrial domains. Its long-standing reputation guarantees a reliable and well-tested platform.
- **Standard Adherence**: Protégé is renowned for its robust support for ontology standards, especially OWL. This ensures that the ontologies developed are interoperable and

can be seamlessly integrated with other systems.

- **Extensibility**: The plugin architecture of Protégé allows users to augment the functionality of the software. The software boasts a vast ecosystem of plugins, ranging from visualisation tools to reasoners such as Pellet and HermiT.
- **Active Community**: The vibrancy and supportiveness of the Protégé community, including forums, mailing lists, and workshops, ensure that assistance is always at hand.
- **Visualisation and Interface**: Protégé's user-centric interface presents a visual representation of ontologies, streamlining the comprehension and management of intricate relationships and hierarchies.
- **Reasoning Capabilities**: With seamless integration capabilities and a myriad of reasoning skills, Protégé is adept at ontology validation, consistency checks, and inferencing, which are paramount for ensuring the accuracy and robustness of the schema layer.
- **Free and Open Source**: The open-source nature of Protégé offers unparalleled transparency and flexibility, coupled with the potential for customisation. This also ensures a transparent cost structure devoid of hidden charges or licensing complications.
- **File Storage and Updates**: For this research, Protégé was employed to craft the proposed ontology model, which was stored as a .ttl file. This format facilitated frequent updates to the ontology model based on emerging information.

The amalgamation of its feature-rich environment, dynamic community, and strict adherence to standards renders Protégé the quintessential choice for constructing the schema layer of the knowledge graph in this study.

7.2.2 Development Process

In robotic assembly cell reconfiguration, an adaptive semantic model is of paramount importance. However, as described in Chapter 4, encapsulating every evolving concept within this model remains challenging. This is particularly apparent in the realm of reconfigurability, where both software and algorithms are in perpetual evolution.

The knowledge graph was designed using the OCCR model, as depicted in Figure 7.2. Insights from industry experts further refined the model. Protégé, a tool well-regarded in both academic and industry sectors for its robustness in ontology modelling, was employed to generate the OCCR model. The completed model was subsequently saved as a ".ttl" file.

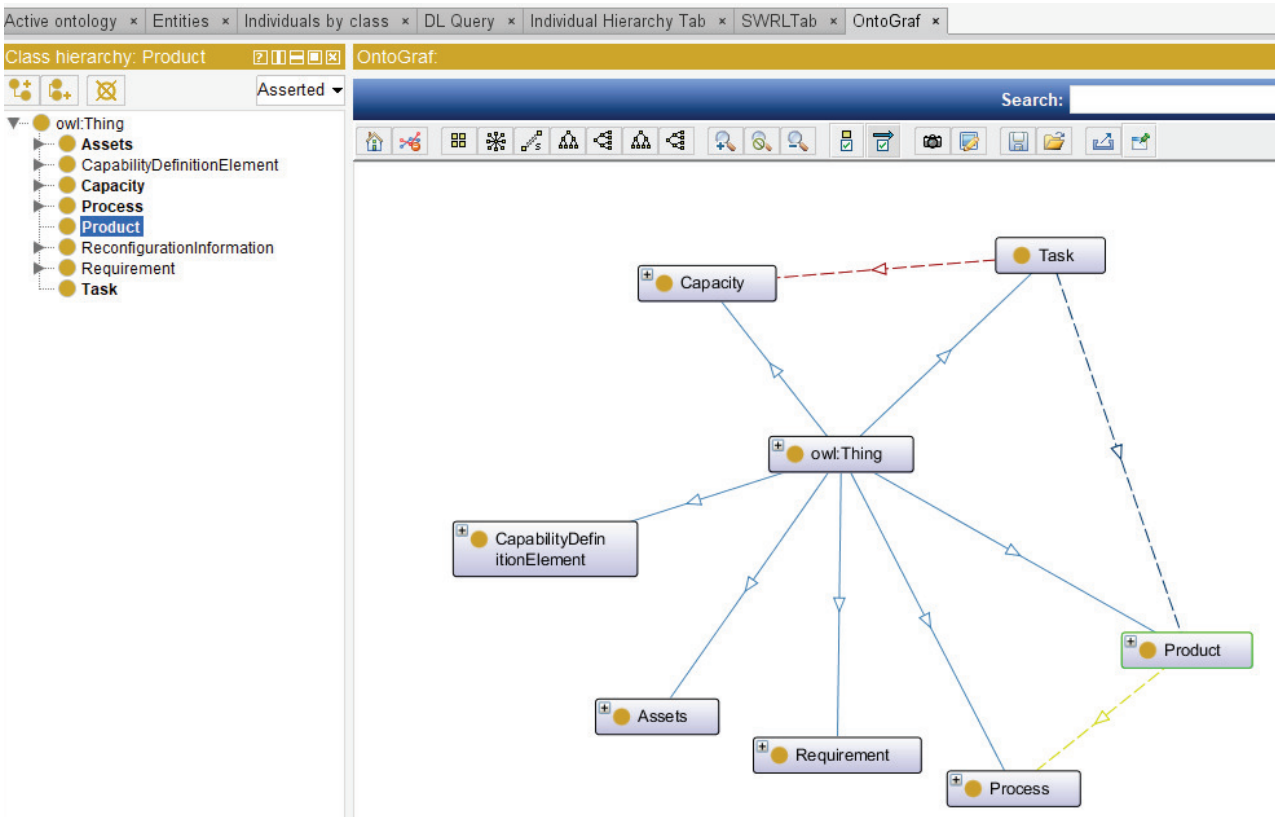


Figure 7.2: Ontology model in Protégé

Thereafter, the entity layer of the knowledge graph was constructed, informed by the schema layer, external data sources, and real-world data. Neo4j was selected for this phase and for the overall generation and storage of the knowledge graph due to its prowess in efficiently managing graph data, the versatility of its query language Cypher, and its scalability [115].

To establish a seamless integration between the Neo4j graph database and the broader programming environment, Py2neo was incorporated. This decision was informed by Py2neo's ability to manipulate the Neo4j database within a Python framework efficiently, streamlining the construction and modification of the knowledge graph in Neo4j [116].

Python was enlisted to oversee various dataset operations, from data loading to ontology model updates and knowledge graph generation. The language's flexibility, vast library ecosystem, and compatibility with various tools made it an ideal choice. Moreover, the Py2neo library,

known for its user interface enhancements and capabilities such as node creation, Cypher query execution, and graph structure updates, was leveraged.

The procedure involving Py2neo and Neo4j is outlined below:

1. **Graph Configurations Initialisation**: Within Neo4j, the `n10s` library, renowned for its RDF data management capabilities, is employed. The `n10s.graphconfig.init` function configures the graph for RDF data, influencing its representation within Neo4j. An example is provided in Figure 7.3.

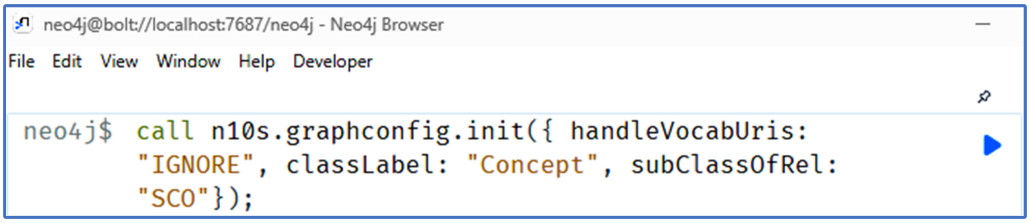


Figure 7.3: Graph initialisation in Neo4j

2. **Ontology Model Import**: After the ontology model is saved as a `.ttl` file, it is imported into Neo4j using commands such as `call n10s.onto.import.fetch`.

3. **Knowledge Graph Construction**: The knowledge graph, encompassing both the schema and entity layers, is built using Py2neo. An example is displayed in Figure 7.4.

4. **Knowledge Graph Refinement**: The knowledge graph undergoes precision enhancements to ensure its accuracy and comprehensiveness.

5. **Knowledge Graph Storage**: Knowledge graphs are archived within Neo4j, ensuring swift and efficient retrieval, with the added advantage of cloud deployment compatibility.

In summary, the software's design was tailored to incorporate tools such as Protégé, Neo4j, Py2neo, and Python. This integration resulted in a powerful platform poised to craft a comprehensive experience databank in the manufacturing sector.

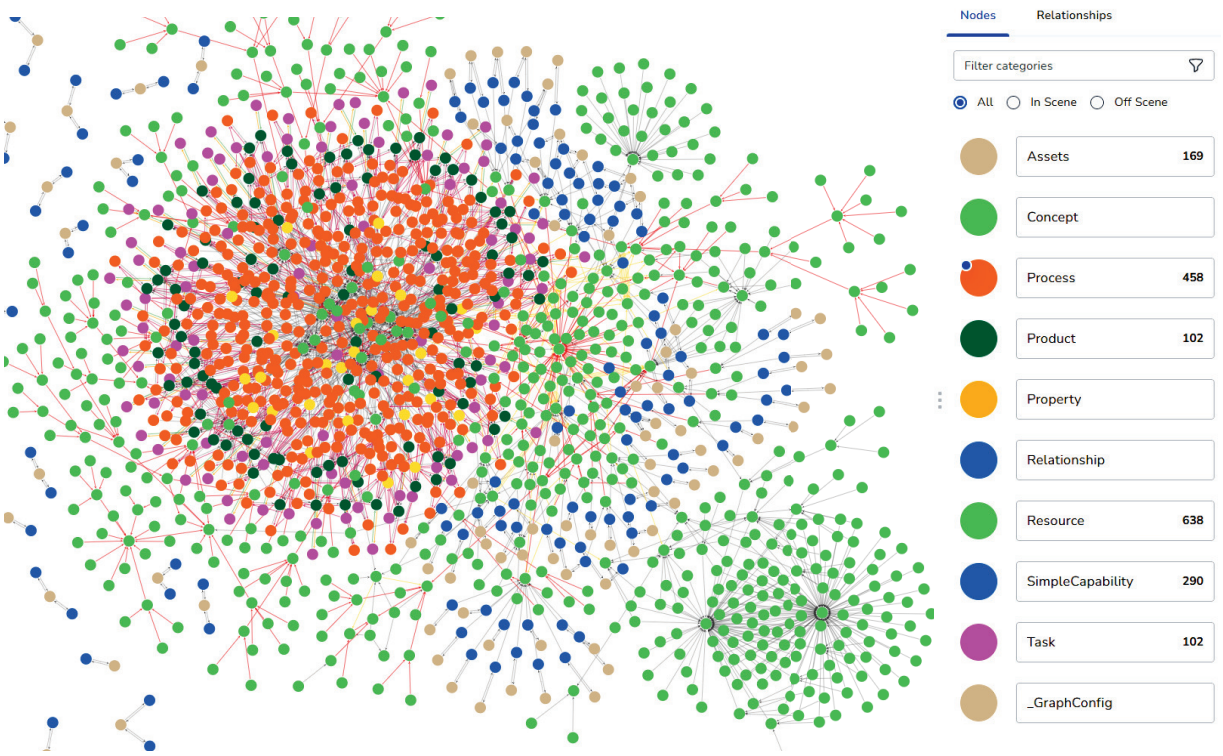


Figure 7.4: Exemplification of a generated knowledge graph

7.3 Interface Development for Asset Selection

Asset selection and evaluation in the domain of advanced manufacturing necessitate an amalgamation of computational and reasoning capabilities. To this end, an interface was developed, leveraging the graph database platform Neo4j and Py2neo, a Python library that interfaces with Neo4j. The overarching goal was to construct an interface anchored on a knowledge graph, ultimately to provide data-driven recommendations and facilitate intricate data management tasks. Figure 7.5 presents a comprehensive UML activity diagram, elucidating the systematic procedure for optimal asset selection and evaluation.

7.3.1 Software Selection for Asset Selection

The development process for the asset selection interface is rooted in the meticulous selection of software tools tailored to address the unique challenges inherent in asset selection in the manufacturing domain. The justification for each tool was based on its individual merits, compatibility, and the specific demands of each phase of the development process.

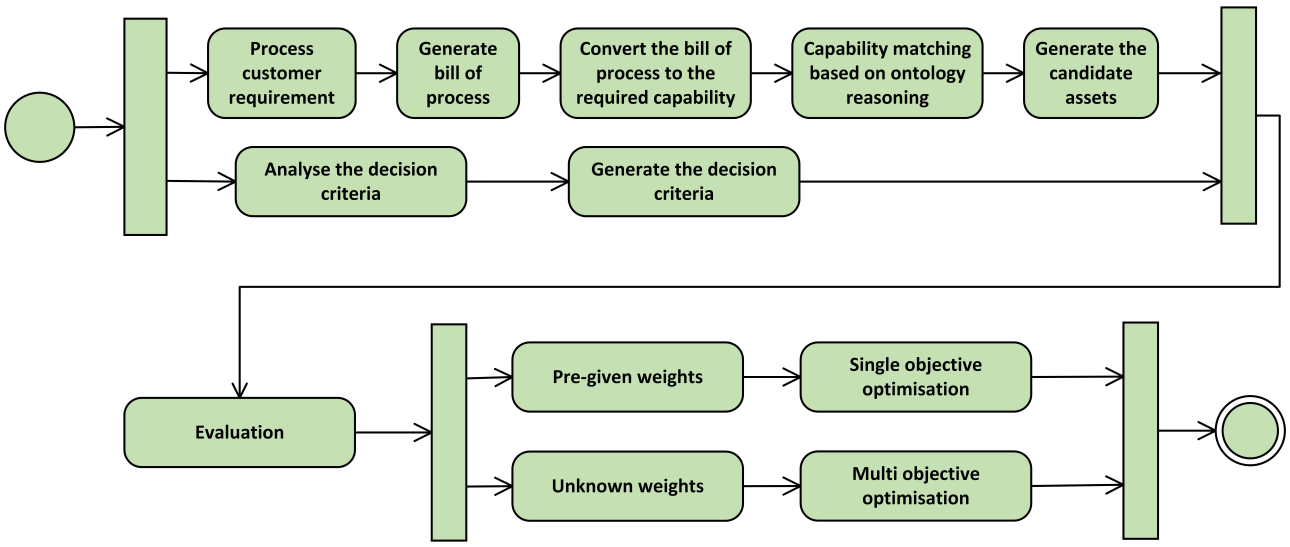


Figure 7.5: UML activity diagram for optimal asset selection

7.3.1.1 Protégé:

Protégé was selected for its proficiency in ontology modelling. Given the ever-evolving nature of manufacturing demands and the continuous advancement of assets and capabilities, an ontology model remains in a constant state of flux. It demands periodic updates, refinements, and augmentations to remain relevant and accurate. The capability of Protégé's capability to conveniently update and modify ontologies earmarks it as the tool of choice for this purpose.

7.3.1.2 Neo4j:

Neo4j's preeminence in graph database management was indispensable. Its powerful query language, Cypher, played a pivotal role in the capability-matching phase.

7.3.1.3 Python and Py2neo:

Python, known for its adaptability and its vast library ecosystem, combined with Py2neo's seamless Neo4j integration, became the backbone of the process. This union permitted intricate data manipulations, evaluations, and subsequent database queries or updates.

7.3.2 Development Process

7.3.2.1 Phase 1: Requirement Gathering and Translation

Upon receiving a customer's specifications, the initial step is to translate these directives into a processable format. This translation involves converting the customer's requisition into a bill of processes, which is subsequently distilled into the requisite capabilities.

7.3.2.2 Phase 2: Capability Matching Using Neo4j

Through Neo4j's Cypher, these capabilities are juxtaposed against available assets in the knowledge graph. The provided Cypher query illustrates how specific tasks and their prerequisites align with potential assets that satisfy those demands, culminating in a compilation of candidate assets aptly poised to cater to the task's requirements as shown in Figure 4.10.

7.3.2.3 Phase 3: Candidate Asset Evaluation

With the candidate asset ensemble collated, the next step is their appraisal. Python's computational prowess is harnessed at this juncture. Each asset undergoes a rigorous evaluation predicated on various metrics such as specification efficiency score, reconfiguration costs, and other relevant parameters. The evaluation process can be bifurcated into:

1. **Single-Objective Optimisation:** When the weights of the decision metrics are predetermined, multi-objective dilemmas can be transformed into a single-objective framework, simplifying the prioritisation of candidate assets.
2. **Multi-Objective Optimisation:** In scenarios bereft of pre-defined weights, Python utilities manifest a Pareto front, elucidating a spectrum of optimal alternatives. This offers decision-makers a myriad of assets, empowering them with the autonomy to make careful selections based on their bespoke needs and strategic preferences. In this scenario, Pymoo [117] is used to enable multi-objective optimisation.

7.3.2.4 Phase 4: Integration and Feedback Loop

The symbiosis between Python and Neo4j, catalysed by Py2neo, ensures a cyclical feedback mechanism. Evaluation outcomes, along with any nascent data or insights, are reincorporated into the database. This not only rejuvenates the knowledge graph but also fosters an environment in which the system can evolve and adapt over time.

7.3.2.5 Conclusion

The software development trajectory for the asset selection interface epitomises the synergistic amalgamation of diverse tools to navigate intricate challenges. With Neo4j orchestrating capability matching, Python supervising asset evaluation, and Protégé ensuring that the ontology remains dynamic and contemporaneous, the system emerges as both resilient and malleable. This development blueprint has been meticulously architected to champion accuracy, efficiency, and versatility, priming it for validation against tangible industry use cases in subsequent design courses.

7.4 Implementing Layout Optimisation

This section details the interface development for the modular optimisation environment, focusing on layout configuration. A UML activity diagram of this framework is depicted in Figure 7.6.

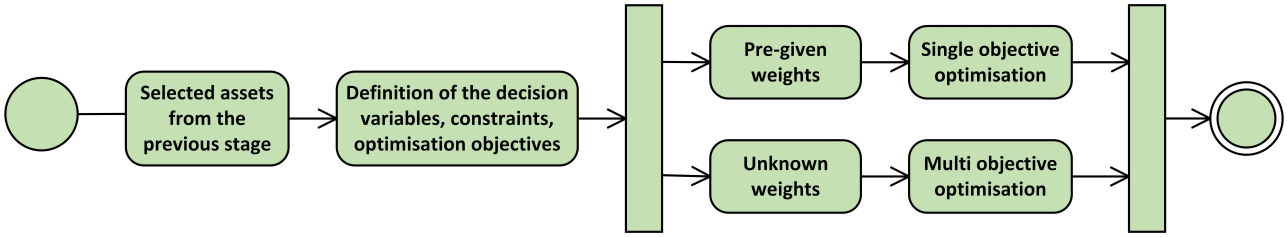


Figure 7.6: UML activity diagram for layout configuration optimisation

7.4.1 Software Selection for Layout Optimisation

As detailed in Chapter 6, the layout optimisation framework is composed of three main components: the simulation environment, the optimisation environment and modular optimisation algorithms. The following sections provide an in-depth explanation of each of these components.

7.4.2 Simulation Environment Interface Development

7.4.2.1 Selecting Appropriate Tools

The selection of simulation tools was informed by the specific requirements of the framework, with an emphasis on broad coverage, versatility, and ease of integration with the Layout optimisation framework.

On the one hand, the Tecnomatix Process Simulate was identified as the most suitable choice for the optimisation of multiple asset layouts. This industry-standard software excels at simulating intricate manufacturing processes at a production-line level. Its comprehensive feature set is tailored for simulating elaborate process flows, making it ideal for large-scale and multifaceted manufacturing scenarios.

On the other hand, RoboDK emerged as the tool of choice for scenarios focused on a single robot, especially those requiring pose optimisation, such as determining optimal operation points. It offers a streamlined and user-friendly interface tailored for simulating specific robotic tasks and assessing various robot models. RoboDK's extensive library of robot models further bolsters the versatility of the framework.

To clarify, in any given layout optimisation scenario, only one of these software tools is employed, ensuring that the most appropriate tool is utilised for the task at hand.

7.4.2.2 Integration and Implementation

The integration of these tools into the framework involved creating bespoke interfaces for each, ensuring their seamless interaction with the layout optimsiation framework. The Tecnomatix

.NET API was used to develop custom applications within the Tecnomatix Process Simulate environment, automating tasks, facilitating data exchange, and executing operations. RoboDK's Python API was leveraged to establish a direct interface with the Python-based optimisation engine, allowing for real-time control and feedback from the simulated robots and environment.

Both software packages underwent thorough testing within the framework to confirm their compatibility and effectiveness. The result is a comprehensive simulation environment that can address various industrial robot manufacturing scenarios, provide insightful feedback for the optimisation process, and handle various robot types and setups. Thus, it is a powerful tool that reduces development time and costs while enhancing efficiency and effectiveness.

7.4.2.3 Development of Simulation Environment in Tecnomatix Process Simulate

The simulation environment within Tecnomatix Process Simulate is pivotal for achieving an optimised manufacturing layout. This section elucidates the connection and synergies between the simulation environment and the optimisation framework, with C# serving as the link between the Python-based optimisation environment and the simulation space.

7.4.2.3.1 Harnessing the Tecnomatix .NET API Tecnomatix offers a .NET API, a bridge for digital manufacturing integration. Two primary interfaces, namely the Command and the Viewer interface, are at the core of this API:

1. **Command Interface:** This interface facilitates task-specific functionalities and allows for the inclusion of user-defined tasks, extending the software's operational range.
2. **Viewer Interface:** This interface provides a continuous graphical user interface (GUI) essential for intuitive data visualisation and manipulation within the Tecnomatix space.

7.4.2.3.2 Constructing the Simulation Space The Tecnomatix .NET API was employed to develop a bespoke library tailored for layout optimisation. The development process encompassed the following:

1. Formulating code within the Visual Studio environment

2. Establishing a GUI interface in Tecnomatix Process Simulate

3. Crafting a dedicated simulation environment

The persistent and interactive nature of the Viewer interface made it the preferred choice for the optimisation task.

7.4.2.3.3 Communication Pathway Design Direct communication between the optimisation and simulation environments is not inherently feasible. C# acts as an intermediary, ensuring a streamlined communication process between the Python optimisation environment and the Tecnomatix Process Simulate, achieved via a socket connection.

7.4.2.3.4 Optimisation Framework Enhancement The optimisation framework introduced in Tecnomatix consists of the following:

1. **Viewer Interface:** An embedded viewer in Tecnomatix Process Simulate.

2. **Command Modules:** Specific commands for defined tasks.

3. **Movement Function:** Facilitating object movement within the simulation.

4. **Signal Detection:** A tool coordinating with the collision detection signal.

In essence, the use of the Tecnomatix .NET API, paired with a structured communication methodology, refines the simulation environment, ensuring a seamless and effective user experience.

7.4.2.4 Simulation Environment Development in RoboDK

RoboDK, which is dedicated to robot simulation and programming, was employed for its inherent advantages, notably its Python API support. This direct support simplifies the integration process, reducing the requirement for middleware platforms, which is a necessity with Tecnomatix Process Simulate.

7.4.2.4.1 Exploiting RoboDK's Functionalities for the Framework RoboDK's attributes, including its Python API, robot model library, and advanced visualisation tools, were harnessed to craft a user-centric simulation environment. The Python API's direct nature streamlines robot programming and control, allowing for precise pose optimisation tasks. Additionally, RoboDK's library negated the need for independent model derivations, enhancing research efficiency.

7.4.2.4.2 RoboDK's Role in the Simulation Process While Tecnomatix Process Simulate excels at multi-faceted simulation scenarios, RoboDK is particularly adept at specific tasks, especially single robot pose optimisation tasks. This distinction underscores the strategic selection of tools based on the simulation's precise needs.

In summary, the integration of both Tecnomatix Process Simulate and RoboDK within the proposed framework ensures a holistic, efficient, and precise simulation environment catering to a diverse range of scenarios.

7.4.3 Optimisation Environment

The optimisation environment is the backbone of the proposed framework. It interacts directly with the simulation environment to perform iterative optimisation tasks. It is also responsible for integrating the optimisation objective sets and optimisation algorithms into the optimisation process. The optimisation environment was developed in Python, leveraging its comprehensive data handling capabilities and extensive library support for optimisation tasks. The environment was designed to be modular and extensible, with each component – the optimisation objective sets, the optimisation algorithms, and the interfaces with the simulation environment and the knowledge graph – developed as separate modules. This modular design ensures that the environment can be easily modified or extended as needed.

In the algorithm, the optimisation objective provides quantitative measures of performance, which are crucial for the optimisation tasks. The framework includes a collection of optimisation objective sets tailored to various manufacturing scenarios. These optimisation objective sets, used as input to the optimisation environment, define the criteria for successful optimisation tasks.

The optimisation objective sets were developed in Python, and Py2neo is used to interact with the Neo4j graph database. This allows for optimisation objective sets to be retrieved from the Manufacturing Reconfiguration Knowledge Graph, applied to the optimisation tasks, and updated with new data as necessary.

Additionally, Neo4j's Cypher Query Language is utilised to interact with the graph database. Users are able to retrieve suggested optimisation objective sets for a particular problem from the graph database. The graph database can subsequently be updated with new information gleaned from recent optimisation tasks.

This continuous data capture, updating, and usage establishes a cyclical learning process. As the system undertakes more optimisation tasks, it amasses a growing wealth of data that can be analysed and used to refine future operations. This ability to learn and improve with each cycle makes the optimisation framework robust and efficient, potentially saving significant resources and enhancing productivity in the long run.

7.4.4 Optimisation Algorithm

The choice of optimisation algorithm is an important aspect of the optimisation environment. The optimisation algorithm determines how the optimal solution to a given problem is found. Some algorithms work better for certain types of problems than others. Therefore, a comprehensive collection of optimisation algorithms has been included in the framework, ensuring a high degree of flexibility and adaptability to diverse optimisation scenarios. The flowchart of utilising the optimising algorithm is depicted in Figure 7.7.

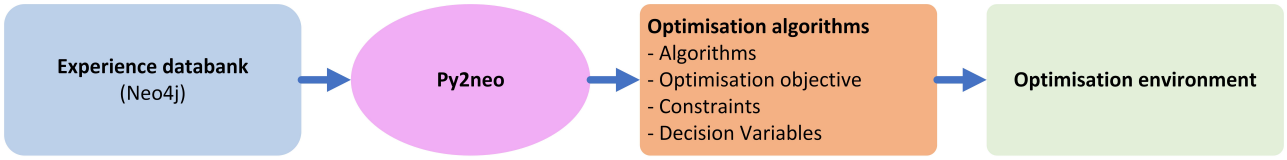


Figure 7.7: Flowchart of using algorithms suggested from experience databank

The integration of these algorithms with the Neo4j graph database was key in the development of this component. Each algorithm was coded in Python, and the Py2neo was utilised to interact with the graph database, retrieving data, executing the optimisation algorithm, and then updating the graph database with the new data.

The algorithms' performance metrics, along with the optimisation objective, decision variables, and constraints utilised in the optimisation tasks, are also stored in the experience databank. This enables users to draw upon the lessons from past optimisations to inform future tasks, facilitating continuous improvement and learning.

7.5 Chapter Summary

This chapter delineates the crucial transition from theoretical models to their practical embodiments, marking a strategic move towards accomplishing Objective 4. This objective, while significantly advanced within the current discussions, awaits its full completion, which is anticipated to occur with the validation of use cases in Chapter 8.

The narrative commenced with an exploration of the experience databank software, a central repository engineered to encapsulate and organise knowledge spanning various tasks, processes, assets, and their reconfiguration. The interface for this databank, discussed in Section 7.2, is underpinned by ontology models and knowledge graph methodologies to facilitate robust information retrieval and utilisation.

Further, the chapter addressed the translation of the asset selection methodology into its software counterpart in Section 7.3. This software instrument employs knowledge graphs alongside multi-criteria decision-making algorithms, representing a sophisticated synthesis tailored for complex asset selection challenges.

Section 7.4 revealed the approach to software-driven layout optimisation. It underscored the convergence of knowledge graphs with simulation tools and artificial intelligence, all calibrated to mesh with industrial standards set by platforms such as Tecnomatix Process Simulate and RoboDK.

While the chapter effectively bridges the gap between earlier theoretical discussions and their practical applications, it also sets the stage for the impending elucidation of real-world use cases. These cases are poised to provide the empirical substantiation needed to satisfy Objective 4 in its entirety, as will be expounded upon in Chapter 8.

Chapter 8

Use Case Studies

This chapter, in conjunction with Chapter 7, contributes to the validation of Objective 4, as set out in this thesis. It introduces three use cases that serve to test and substantiate the methodologies developed. The use cases addressed are as follows:

1. Use Case 1 focuses on executing asset selection and system layout optimisation based on a single objective with the help of the experience databank.
2. Use Case 2 explores the extension of layout optimisation to single-robot operations, emphasising multi-criteria decision-making.
3. Use Case 3 assesses modular and multi-objective asset selection using the experience databank, expanding the asset selection validation to include multi-criteria and modularity aspects.

8.1 Use Case 1 – Asset Selection and System Layout Optimisation Using the Experience Databank

Use Case 1 is introduced to validate the methodology for building and updating the experience databank, optimal manufacturing asset selection, and system layout optimisation as shown in Figure 8.1.

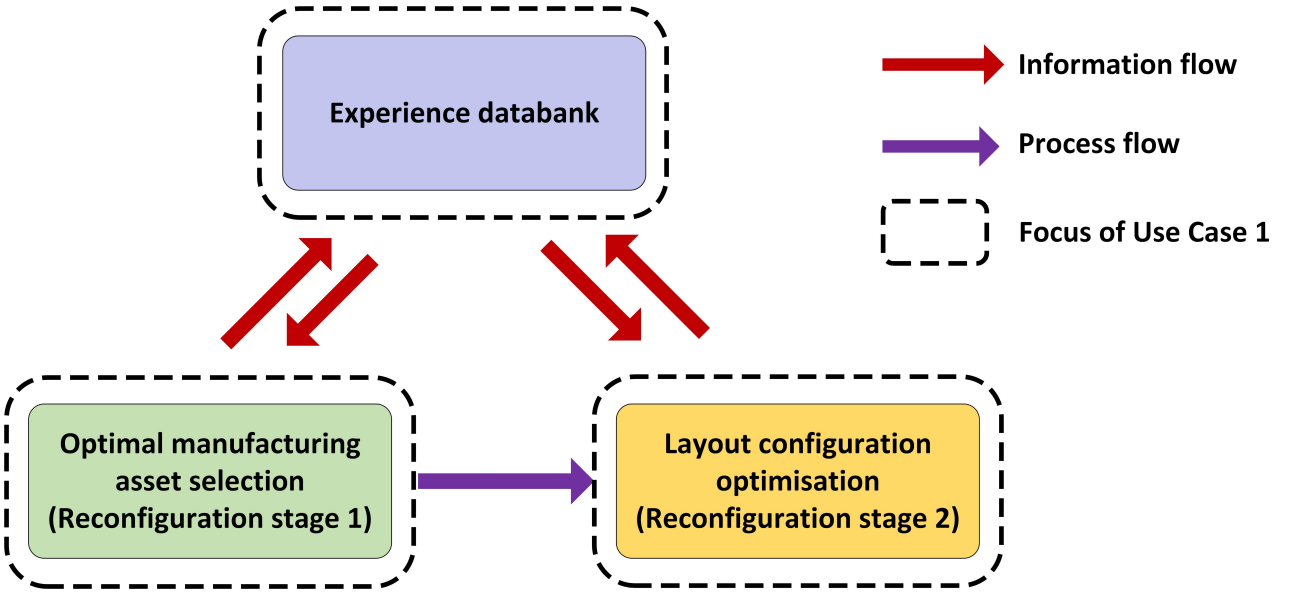


Figure 8.1: Validation components of the proposed methodology in Use Case 1

8.1.1 Introduction

The validation procedure was carried out using a dataset procured from the OMNIFACTORY demonstrator at the University of Nottingham [118], as illustrated in Figure 8.2. OMNIFACTORY, a state-of-the-art £3.8 million facility, is a national testbed for intelligent manufacturing systems in the UK. Its mission is to expedite accurate shop floor reconfigurations to meet customers' bespoke requirements. Located on the University's Jubilee Campus, OMNIFACTORY is fitted with a bespoke flooring system that enables a unique, reconfigurable environment. Nonetheless, determining the optimal configuration for a new or altered product presents a considerable challenge.

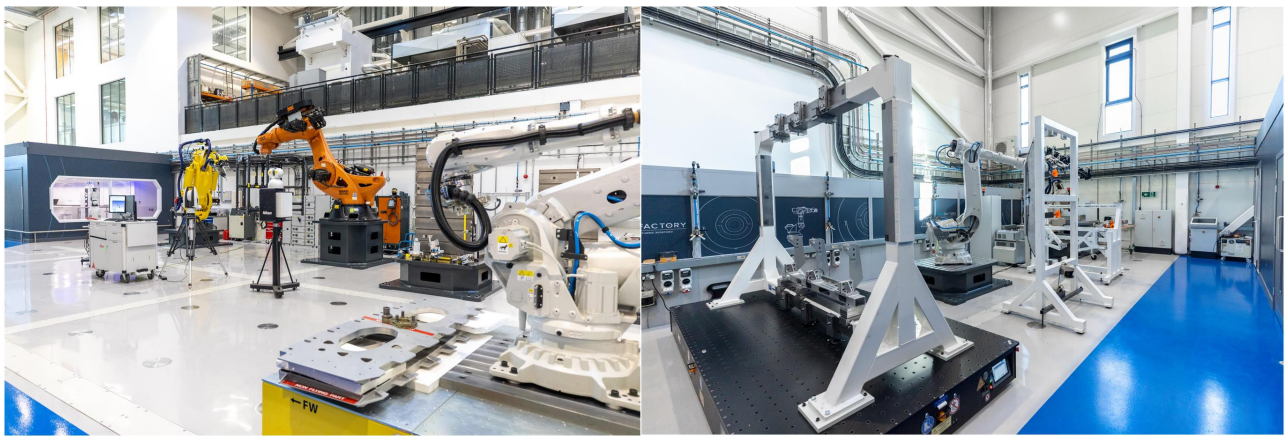


Figure 8.2: Physical layout of one of the OMNIFACTORY's test plants

The validation dataset was compiled using technical documents, equipment data, and product

design files obtained from OMNIFACTORY and its partner organisations. It includes 101 tasks – related and unrelated to reconfiguration – along with their respective requirements and procedures for each task. The dataset features 161 potential assets with comprehensive details to facilitate capability matching and reconfiguration. These assets consist of production line components and resources from the asset pool, such as hardware, software, a human workforce, and reconfiguration solvers to enhance production and reconfiguration activities. To maintain confidentiality, all information was anonymised: task names were altered to "Task 1", "Task 2", and so on; customer names were altered to "Company A", "Company B", and so on; and product features were replaced with generic terms such as "Feature 1" and "Feature 2".

This case study emphasises the practical deployment of the methodology within an authentic industrial setting and its effectiveness in navigating the reconfiguration process by leveraging the attributes of knowledge graphs to handle data diversity.

8.1.2 Construction of the Knowledge Graph

In this current use case, the experience databank is built based on the implementation described in Chapter 7. Both top-down and bottom-up approaches are utilised. The OCCR model is used as the schema layer, and Neo4j is used to build the knowledge graph.

Figure 8.3 showcases the generated knowledge graph featuring explicit relationships. To enhance and complete the knowledge graph, an ontology-rule-based methodology is utilised, as expounded in the work of Chen et al. [104]. An illustrative example is when an entity from the task node is connected to an entity from the product node, and the corresponding entity from the product node is linked with an entity from the process node through the "requiredProcess" relationship. A new relationship is then generated, named "taskRequiredProcess" which connects the entity from the task node to the entity from the process node. This enhanced knowledge graph can be employed to augment the reconfiguration process. The generated knowledge graph is the implementation of "Experience Databank".

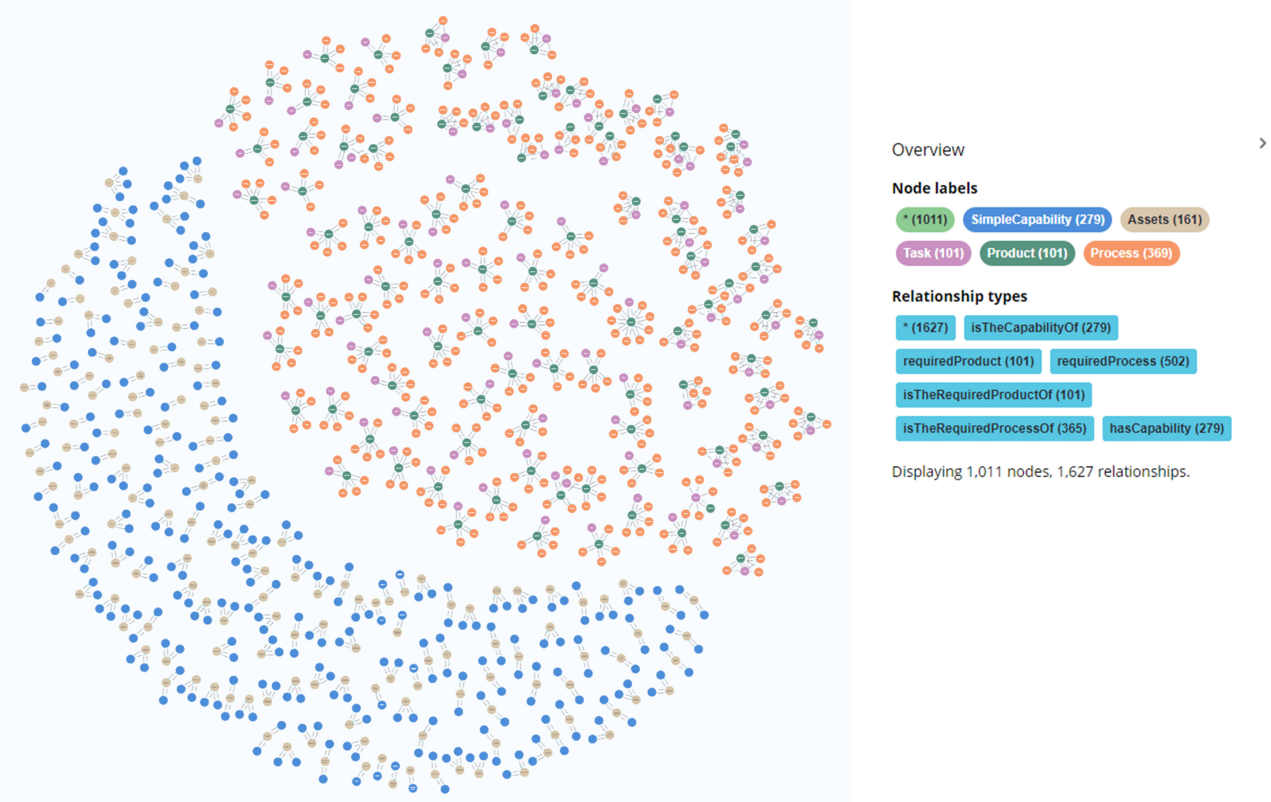


Figure 8.3: Construction of the knowledge graph (experience databank)

8.1.3 Application of the Experience Databank

Once the experience databank is established, its effectiveness and the improvements it brings to the reconfiguration process can be demonstrated through two specific applications. The first application addresses the selection of resources for a task characterised as non-reconfiguration-related, while the second application relates to the provision of reconfiguration recommendations for a reconfiguration-related task.

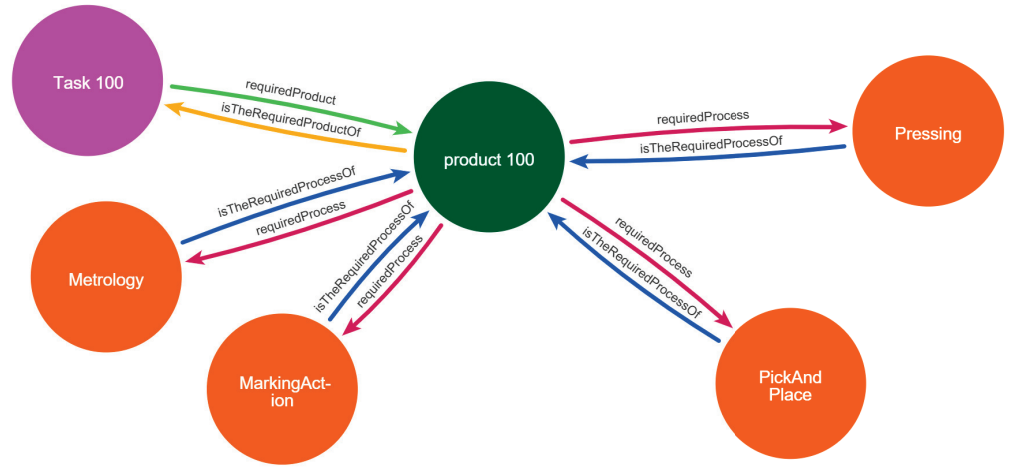
8.1.3.1 Application 1: Resource Selection for Non-Reconfiguration-Related Task

In the first application, "Task 100" – a non-reconfiguration-related task – in the knowledge graph is selected as the demonstration task. The experience databank is used to help the task find the most appropriate assets. This process consists of two steps: finding the required process to produce the product from "Task 100" (Step 1) and capability matching between the required process and the available capability to find the candidate assets based on the capability information (Step 2). This example demonstrates not only capability matching based on ontology reasoning but also how to decompose the combined capability to find the

potential assets for the associated input capability. For the first step, the required process for "Task 100" can be determined via the query command in Neo4j, as shown in Figure 8.4. The implemented methodology reveals that Product 100 had two distinct features. Figure 8.5 demonstrates that feature 1 necessitates the "PickAndPlace" and "MarkingAction" processes, while feature 2 required the "Pressing" and "Metrology" processes.

```
1 MATCH (a:Task {Name:"Task 100"})-[:requiredProduct]→
  (b:Product)-[:requiredProcess]→(c:Process)
2 RETURN a,b,c
```

Figure 8.4: Query command to find the required process for "Task 100" in Neo4j



Process **MarkingAction**

Properties | Neighbors | Relationships

Edit

Process

feature	F1
MaximumCost	1500
MaximumUtilizationRate	0.4
ProcessName	MarkingAction
ProcessSequence	2
ProductName	product 100
requiredAccuracy	0.05
requiredResolution	500

Process **PickAndPlace**

Properties | Neighbors | Relationships

Edit

Process

feature	F1
MaximumCost	4400
MaximumUtilizationRate	0.5
ProcessName	PickAndPlace
ProcessSequence	1
ProductName	product 100
requiredAccuracy	0.51
requiredGraspingForce	300
requiredGraspingType	0
requiredPayload	200
requiredReachbility	3000

Figure 8.5: Information about task 100 and its related nodes in the knowledge graph (experience databank)

In relation to the capability-matching process (Step 2), Figure 8.5 also illustrates that the

requirements for various processes within the task can be queried using the generated knowledge graph. For example, for Feature 1, "PickAndPlace" and "MarkingAction" are the necessary processes. In the OCCR model, the "MarkingActionCapability" is categorised as a simple capability, whereas the "PickAndPlaceCapability" is recognised as a combined capability. This distinction allows for a demonstration of how the model operates with varied capability types through the capability-matching process. Specifically, the matching process for the "MarkingAction" and the "PickAndPlace" processes can be demonstrated.

The specification requirement of "MarkingAction" is "requiredAccuracy: 0.05" and "requiredResolution: 500". The capacity requirement is the cost and utilisation rate. The allowed maximum cost for "MarkingAction" is 1,500, and the allowed maximum utilisation rate is 0.4. The ontology reasoning approach is used in the use case to find the potential assets based on the requirements and specifications and the capacity information. As shown in Figure 8.6, with this ontology reasoning approach, not only the related assets for the "MarkingActionCapability" but also the assets with the capability of the subclass of "MarkingActionCapability" can be found. In this figure, SCO means "subclass of". The subclasses of the "MarkingActionCapability" are "LaserMarkingCapability", "InkMarkingCapability", "PrintingCapability", "StampMarkingCapability", and "LabellingCapability". The related assets for these subclasses can be found automatically without extra effort based on the ontology information. The knowledge inference process (ontology reasoning) in Neo4j is essential because it allows for the representation and manipulation of complex relationships between entities in a graph database. Using ontologies or formal models of a particular domain, the system can automatically deduce new information (in this case, the subclasses of the "MarkingActionCapability") based on the rules encoded in the ontology. This can improve data accuracy, consistency, and completeness and help users make more informed decisions.

The results of the capability-matching process are presented in Figure 8.7. For the "MarkingAction" process in Task 100, six assets are identified to satisfy the requirement according to the capability matching process based on the specification requirement (accuracy and resolution) without considering the capacity information. The capacity model is employed to improve the resource selection process. In the study, cost and utilisation rate are used to represent the capacity information of the candidate assets. Only "InkMarker-3" fulfils all the requirements and is thus chosen.

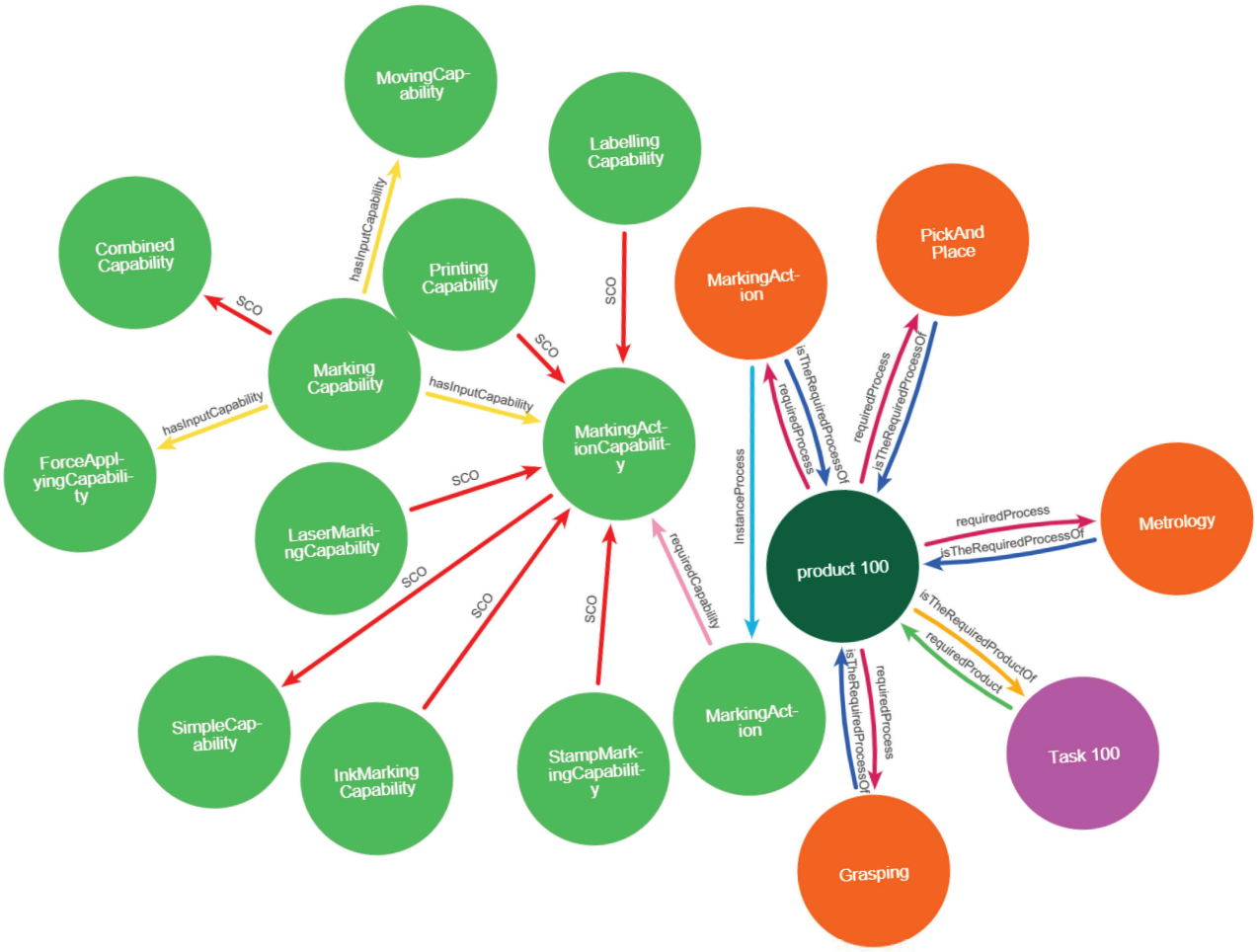
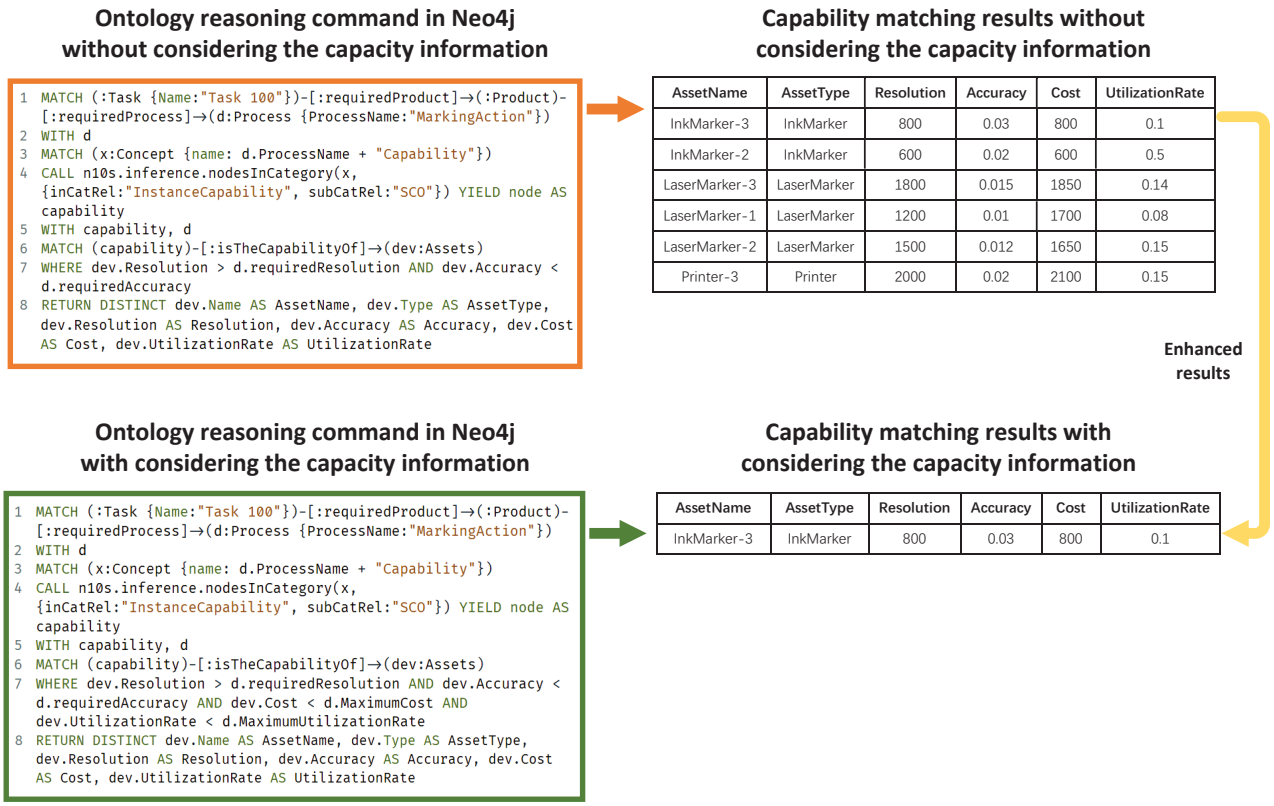


Figure 8.6: Capability matching for "MarkingCapability" with the help of ontology reasoning

Regarding the execution of the process for "PickAndPlace", Figure 8.5 indicates that the specifications for "PickAndPlace" process include "requiredPayload: 200", "requiredGraspingForce: 300", and "requiredReachability: 3,000". The capacity requirements include "MaximumCost: 4,000" and "MaximumUtilizationRate: 0.5". Compared with the capability matching process for a simple capability, this process has an extra process called capability decomposition. Since the "PickAndPlace" capability is a combined capability, it must be decomposed. The decomposed capability is identified based on ontology reasoning, as illustrated in Figure 8.8.

It can be observed that "Moving", "Releasing", "Grasping", and "ForceApplying" are the decomposed capabilities (input capabilities). In the OCCR model, "Moving" and "ForceApplying" are two simple capabilities of the robots. Hence, only one robot is required to execute these two capabilities. The same applies to "Releasing" and "Grasping", where only one gripper is necessary to execute these two capabilities. The capability-matching results, as presented in Figure 8.9, were obtained using ontology reasoning.




```
MATCH (:Task {Name:"Task 100"})-[:requiredProduct]→(:Product)-
[:requiredProcess]→(d:Process {ProcessName:"MarkingAction"})
WITH d
MATCH (x:Concept {name: d.ProcessName + "Capability"})
CALL n10s.inference.nodesInCategory(x,
{inCatRel:"InstanceCapability", subCatRel:"SCO"}) YIELD node AS
capability
WITH capability, d
MATCH (capability)-[:isTheCapabilityOf]→(dev:Assets)
WHERE dev.Resolution > d.requiredResolution AND dev.Accuracy <
d.requiredAccuracy
RETURN DISTINCT dev.Name AS AssetName, dev.Type AS AssetType,
dev.Resolution AS Resolution, dev.Accuracy AS Accuracy, dev.Cost
AS Cost, dev.UtilizationRate AS UtilizationRate
```

AssetName	AssetType	Resolution	Accuracy	Cost	UtilizationRate
InkMarker-3	InkMarker	800	0.03	800	0.1
InkMarker-2	InkMarker	600	0.02	600	0.5
LaserMarker-3	LaserMarker	1800	0.015	1850	0.14
LaserMarker-1	LaserMarker	1200	0.01	1700	0.08
LaserMarker-2	LaserMarker	1500	0.012	1650	0.15
Printer-3	Printer	2000	0.02	2100	0.15

```
MATCH (:Task {Name:"Task 100"})-[:requiredProduct]→(:Product)-
[:requiredProcess]→(d:Process {ProcessName:"MarkingAction"})
WITH d
MATCH (x:Concept {name: d.ProcessName + "Capability"})
CALL n10s.inference.nodesInCategory(x,
{inCatRel:"InstanceCapability", subCatRel:"SCO"}) YIELD node AS
capability
WITH capability, d
MATCH (capability)-[:isTheCapabilityOf]→(dev:Assets)
WHERE dev.Resolution > d.requiredResolution AND dev.Accuracy <
d.requiredAccuracy AND dev.Cost < d.MaximumCost AND
dev.UtilizationRate < d.MaximumUtilizationRate
RETURN DISTINCT dev.Name AS AssetName, dev.Type AS AssetType,
dev.Resolution AS Resolution, dev.Accuracy AS Accuracy, dev.Cost
AS Cost, dev.UtilizationRate AS UtilizationRate
```

AssetName	AssetType	Resolution	Accuracy	Cost	UtilizationRate
InkMarker-3	InkMarker	800	0.03	800	0.1

Figure 8.7: The capacity model enhances the resource selection process

The candidate assets for the "ForceApplying" and "Moving" capability, without considering the capacity information, are *FANUC M-900iB/400L, KUKA KR210 R3300 K ultra, KUKA KR 2100 R3100 ultra, FANUC M-200iA/900L, KUKA KR 240 R3330, KUKA KR 210-2 3100, FANUC M-2000iA/1700L*. Similarly, the candidate assets for the "Grasping" and "Releasing" capabilities were *Vacuum Gripper-6, Vacuum Gripper-5, Finger Gripper-5, Finger Gripper-6, and Finger Gripper-4*. The capacity model is utilised to enhance the capability results, and the cost information considers both the cost of the robot and the gripper. Based on Figure 8.10, two combinations satisfy the requirement, namely [*KUKA KR210 R3100 ultra, Finger Gripper-4*], and [*KUKA KR210 R3100 ultra, Vacuum Gripper-5*]. The candidate assets information from these two combinations is listed in Table 8.1.

Command in Neo4j to find the input capability of PickAndPlace

```
1 MATCH (:Task{Name:"Task 100"})-[:requiredProduct]→
  (:Product)-[:requiredProcess]→
  (d:Process{ProcessName:"PickAndPlace"})
2 WITH *
3 MATCH (x:Concept {name: d.ProcessName +
  "Capability"})
4 CALL n10s.inference.nodesInCategory(x,
  {inCatRel:"SCO_RESTRICTION"}) YIELD node AS f
5 MATCH (x)-
  [r:SCO_RESTRICTION{onPropertyName:"hasInputCapabilit
  y"}]→(f)
6 RETURN f,r,x
```

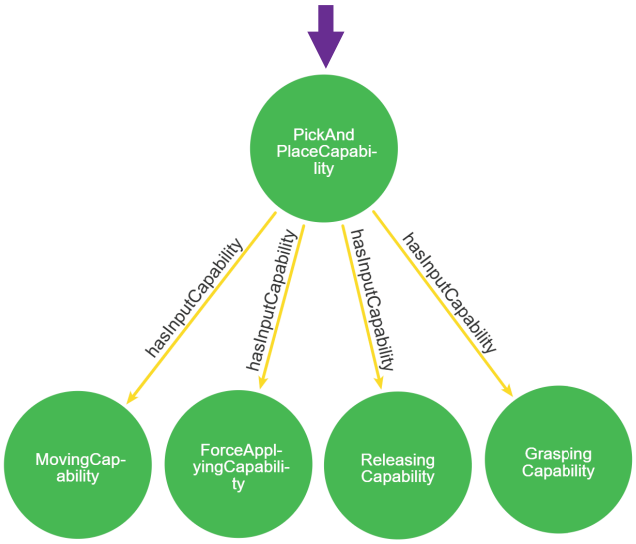


Figure 8.8: Capability decomposition process for PickAndPlace capabiltiy

Table 8.1: Combination asset information

	Assets	Payload	Reachability	Utilisation Rate	Cost	Force
Combination 1	KUKA KR210 R3100 ultra	210	3,301	0.4	4,500	
	Finger Gripper-4				1,000	400
Combination 2	KUKA KR210 R3100 ultra	210	3,301	0.4	4,500	
	Vacuum Gripper-5				1,100	340

If the customer's priority is cost-effectiveness, they should opt for Combination 1, which consists of the *KUKA KR210 R3100 ultra* paired with the *Finger Gripper-4*; however, a different evaluation process will be carried out if the customer seeks an optimal solution that avoids overspecification, which can be referred to the specification efficiency score as described in Section 5.4.2.1. In this case, for Combination 1 and Combination 2, because the robot is the

Finding candidate assets for "ForceApplying" and "Moving" capability

```
MATCH (:Task{Name:"Task 100"})-[:requiredProduct]→(:Product)-
[:requiredProcess]→(d:Process{ProcessName:"PickAndPlace"})
WITH *
MATCH (x:Concept {name: d.ProcessName + "Capability"})
CALL n10s.inference.nodesInCategory(x,
{inCatRel:"SCO_RESTRICTION"}) YIELD node AS capability
MATCH (x)←
[r:SCO_RESTRICTION{onPropertyName:"IsTheInputCapabilityOf"}]-
(capabilityConcept)
WITH d, capabilityConcept
CALL n10s.inference.nodesInCategory(capabilityConcept,
{inCatRel:"InstanceCapability"}) YIELD node AS capability
WITH d, capability
MATCH (capability)-[:isTheCapabilityOf]→
(dev:Assets{Type:"Robot"}) WHERE dev.Payload > d.requiredPayload
AND dev.Reachability > d.requiredReachability
RETURN DISTINCT dev.Name as AssetName, dev.Type as AssetType,
dev.Payload as Payload, dev.Reachability as Reachability,
dev.UtilizationRate as UtilizationRate, dev.Cost as Cost
```

AssetName	AssetType	Payload	Reachability	UtilizationRate	Cost
FANUC M-900iB/400L	Robot	400	3704	0.8	5200
FANUC M-2000iA/1700L	Robot	1700	4683	0.3	6000
KUKA KR 240 R3330	Robot	240	3326	0.4	4500
FANUC M-2000iA/900L	Robot	900	4683	0.3	5500
KUKA KR 210 R3300 K ultra	Robot	210	3301	0.4	4500
KUKA KR 210 R3100 ultra	Robot	240	3095	0.4	3200

Finding candidate assets for "Grasping" and "Releasing" capability

```
MATCH (:Task{Name:"Task 100"})-[:requiredProduct]→(:Product)-
[:requiredProcess]→(d:Process{ProcessName:"PickAndPlace"})
WITH *
MATCH (x:Concept {name: d.ProcessName + "Capability"})
CALL n10s.inference.nodesInCategory(x,
{inCatRel:"SCO_RESTRICTION"}) YIELD node AS capability
MATCH (x)←
[r:SCO_RESTRICTION{onPropertyName:"IsTheInputCapabilityOf"}]-
(capabilityConcept)
WITH d, capabilityConcept
CALL n10s.inference.nodesInCategory(capabilityConcept,
{inCatRel:"InstanceCapability"}) YIELD node AS capability
WITH d, capability
MATCH (capability)-[:isTheCapabilityOf]→
(dev:Assets{Type:"Grippers"}) WHERE
dev.GraspingForce_max>d.requiredGraspingForce or
dev.HoldingForce_max>d.requiredGraspingForce
RETURN DISTINCT dev.Name as AssetName, dev.Type as AssetType,
dev.Subtype as Subtype, dev.GraspingForce_max as GraspingForce,
dev.HoldingForce_max as HoldfingForce, dev.Cost as Cost
```

AssetName	AssetType	Subtype	GraspingForce	HoldingForce	Cost
Vacuum Gripper-5	Gripper	Vacuum Gripper		340	1100
Vacuum Gripper-6	Gripper	Vacuum Gripper		400	1200
Finger Gripper-6	Gripper	Finger Gripper	500		1500
Finger Gripper-4	Gripper	Finger Gripper	400		1000
Finger Gripper-5	Gripper	Finger Gripper	450		1200

Figure 8.9: Finding candidate assets for "ForceApplying", "Moving", "Grasping" and "Releasing" capability

same, only the specification score of Finger Gripper-4 and Vacuum Gripper-5 should be calculated. To calculate this score, the requirement should be known; in this case, the requirement is that the force should be greater than 300. On this basis, the fuzzy priority relations based on Equation (5.5) are built first. According to an assessment of the value of candidate assets, the closer the features of the evaluators of the candidate assets are to the evaluators, the higher the score will be. As the force is the only sub-evaluator, the matrix is defined in Equation (8.1) according to Equation (5.5).

$$B_1 = \begin{bmatrix} 0.5 & 0 \\ 1 & 0.5 \end{bmatrix} \tag{8.1}$$

The consistent fuzzy matrix can be calculated in Equation (8.2) according to Equation (5.10)

and Equation (5.13).

$$R_1 = \begin{bmatrix} 0.5 & 0.25 \\ 0.75 & 0.5 \end{bmatrix} \tag{8.2}$$

The score s_d^1 (where $d = 1, 2$) of the two candidate assets (Finger Gripper-4 and Vacuum Gripper-5) under the factor "force" can be calculated in Equations (8.3) and (8.4) according to Equation (5.10):

$$s_d^1 = 0.3660 \tag{8.3}$$

$$s_d^2 = 0.6340 \tag{8.4}$$

Thus, in this case, combination two (*KUKA KR210 R3100 ultra, Vacuum Gripper-5*) should be chosen because s_d^2 in Equation (8.4) has a higher specification efficiency score.

8.1.3.2 Application 2: Enhancing the Reconfiguration Task with the Semantic Reconfiguration Model

This application demonstrates, through an application from the OMNIFACTORY project at the University of Nottingham, the ability of the reconfiguration model to improve the reconfiguration process. This task is stored in the knowledge graph and marked as "Task 50" – a reconfiguration task in the aerospace domain, as depicted in Figure 8.11. There are two types of processes in the reconfiguration task: (1) the current process, which needs to be reconfigured, and (2) the reconfiguration solution, such as the layout reconfiguration process, resource selection process, and job scheduling process. With these two types of processes, reconfiguration can be better described. Not only information about the current production but also the solutions required to perform reconfiguration are explained.

In Figure 8.12, a frame is prominently positioned on the automated guided vehicles (AGV). While there are three candidate robots, only one will be selected to approach the AGV for a specific assembly task. This robot's primary responsibility is to retrieve the front and aft beams from a beam storage rack and assemble them onto the AGV's frame, preparing it for further parts assembly. Nearby, an end effector tool stand displays two specialised tools: the "Metrology" end effector and the "Pick-and-Place" end effector. Below is a detailed description of the task of this robotic assembly cell.

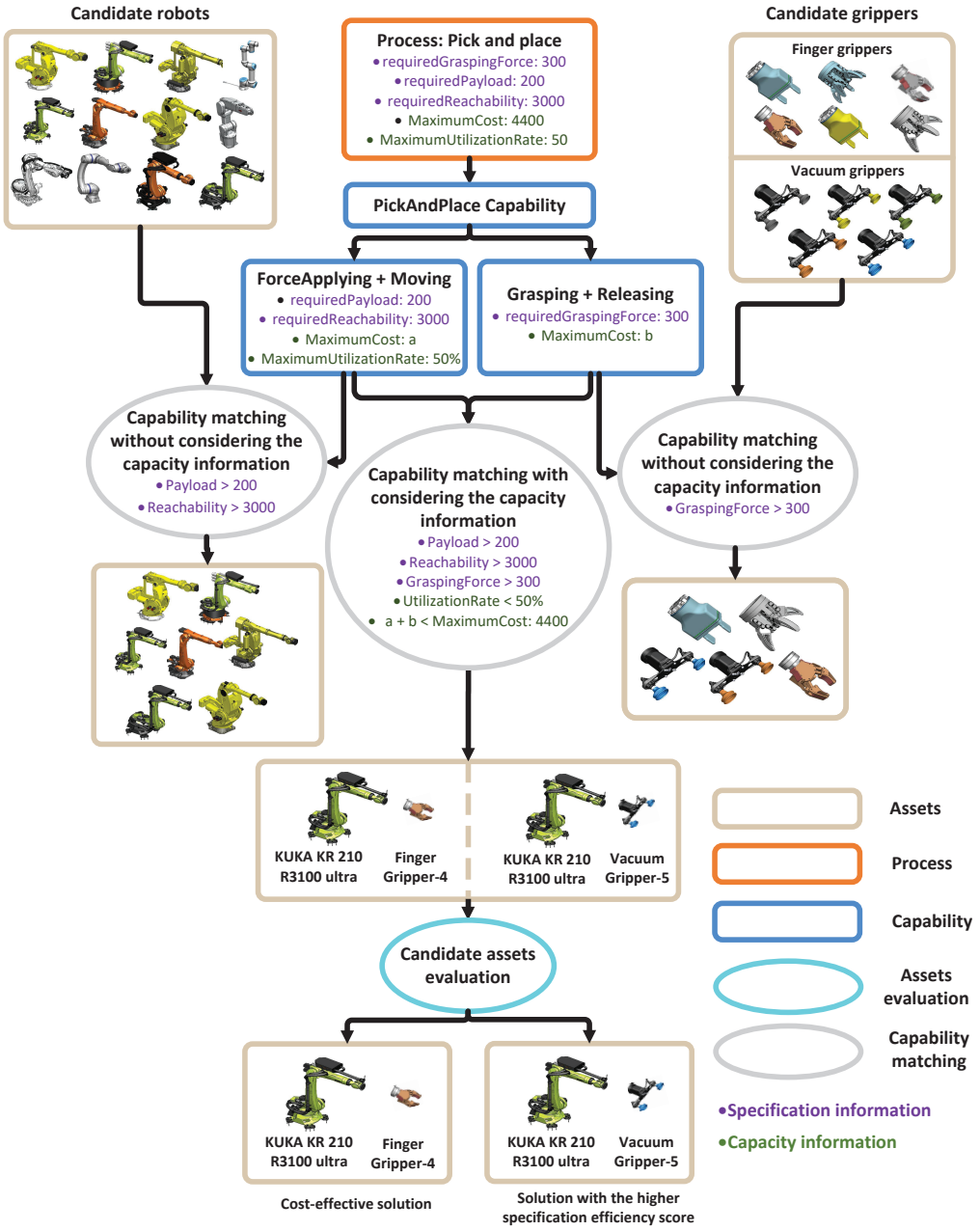


Figure 8.10: Detailed process about finding candidate assets for "ForceApplying", "Moving", "Grasping" and "Releasing" capability

1. **Mount the Pick-and-Place End Effector**

   The selected candidate robot first mounts the pick-and-place end effector to enable the pick-and-place capability.

2. **Pick-and-Place the Front Beam**

   The selected candidate robot picks the front beam and places it on the upper side of the frame.

3. **Pick-and-Place the Aft Beam**

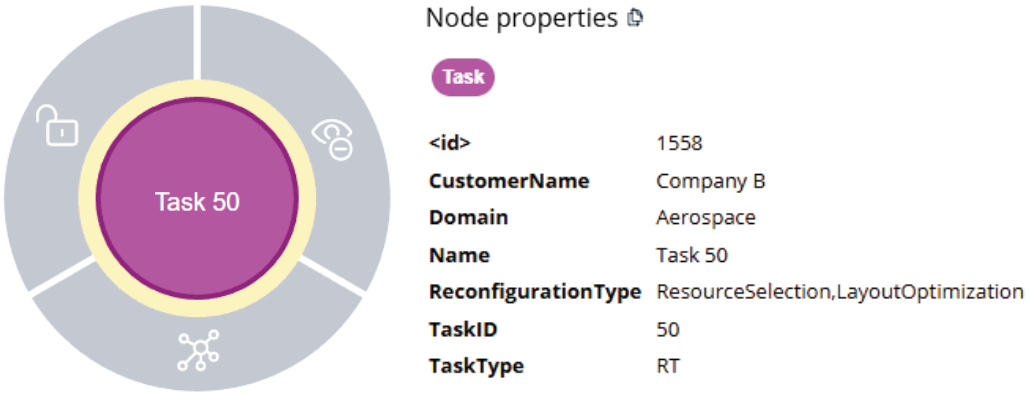


Figure 8.11: Information about Task 50

The selected candidate robot picks the aft beam and places it on the lower side of the frame.

4. **Unmount the Pick-and-Place End Effector**
   The selected candidate robot unmounts the pick-and-place end effector on the tool stand.

5. **Mount the Metrology End Effector**
   The selected candidate robot mounts the metrology end effector to enable the metrology capability.

6. **Execute the Metrology Operation**
   The selected candidate robot utilises the metrology end effector for metrology on the mounted front beam and the after beam.

7. **Unmount the Metrology End Effector**
   At last, the selected candidate robot unmounts the metrology end effector and puts it on the tool stand.

On the other hand, the reconfiguration solution for this task comprises two main components: "resource selection" and "layout optimisation".

Through ontology reasoning in the knowledge graph (experience databank), the resource selection model offers guidance on the objectives, decision variables, constraints, and reconfiguration solutions that should be considered for this task, as depicted in Figure 8.13.

In this case, the specification efficiency score guides the selection of the candidate robots. Based on the experience databank and the engineer's experience, evaluators are defined and

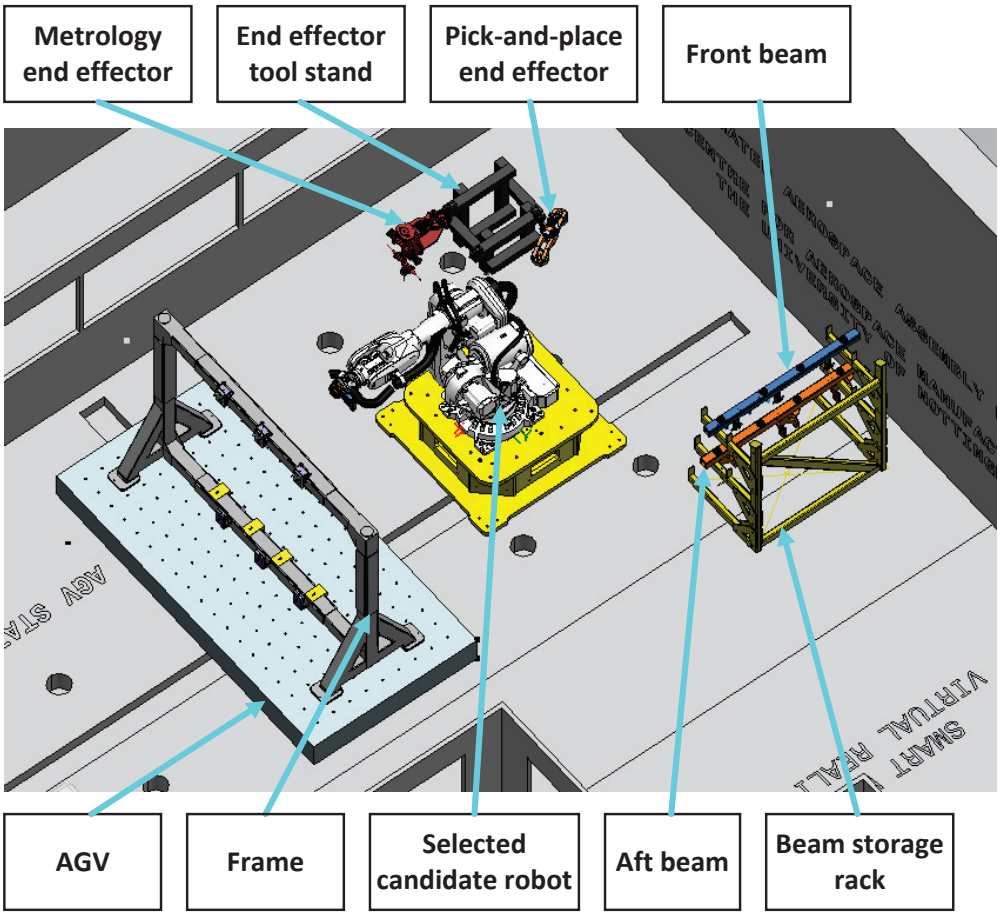


Figure 8.12: Initial physical layout of the production cell from the Task 50

used to select the most suitable assets for the applications. Furthermore, fuzzy evaluation is used to select the optimal resources [119]. The product requirement in the use case is listed in Table 8.2.

Table 8.2: Requirements

Sub-Evaluators	Requirement
Repeatability	> 0.05
Payload	> 100
Reachability	$> 2{,}500$

The deviation between the evaluators and the features from the candidate assets (KUKA KR 1000 titan, ABB IRB 6700-150 robot, FANUC M-900iB/360 robot) is then calculated. Based on the method proposed in Chapter 5, Table 8.3 presents detailed information regarding the candidate assets from the experience databank.

As mentioned in Chapter 5, an evaluation matrix is defined to evaluate the candidate assets. In the validation case, the consistent fuzzy matrix is utilised.

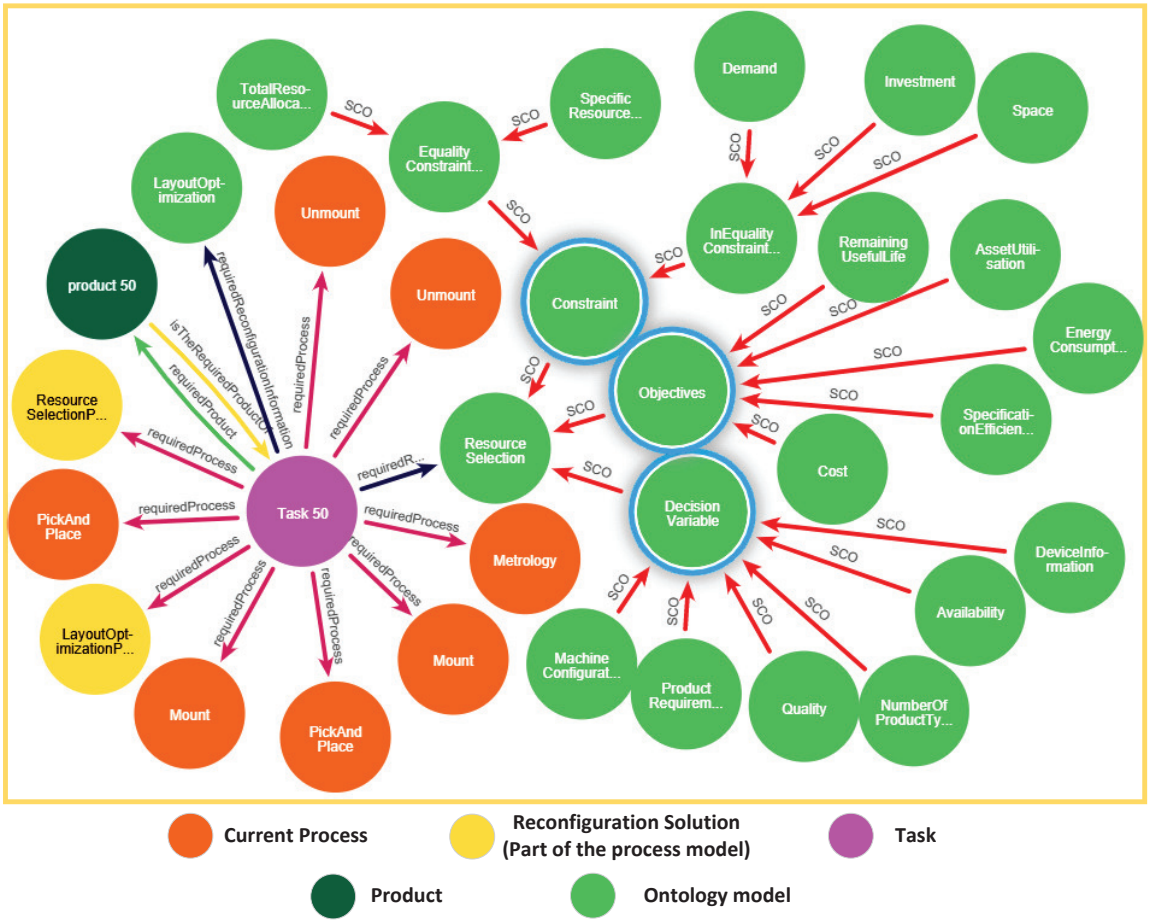


Figure 8.13: Reconfiguration model to enhance the resource selection process

Table 8.3: Candidate asset information

Sub-Evaluators	KUKA KR 1000 titan	ABB IRB 6700-150	FANUC M-900iB/360
Repeatability	0.2	0.06	0.3
Payload	1,000	150	700
Reachability	3,202	3,200	2,832

In this case, the fuzzy priority relation based on Equation (5.5) is built first. Taking "Repeatability" as an example, the matrix is defined using Equation (8.5) according to Equation (5.5).

$$B_1 = \begin{bmatrix} 0.5 & 0 & 1 \\ 1 & 0.5 & 1 \\ 0 & 0 & 0.5 \end{bmatrix} \tag{8.5}$$

The consistent fuzzy matrix is calculated using Equation (8.6) according to Equations (5.10) and (5.13).

$$R_1 = \begin{bmatrix} 0.5 & 0.333 & 0.667 \\ 0.667 & 0.5 & 0.833 \\ 0.333 & 0.667 & 0.5 \end{bmatrix} \tag{8.6}$$

The score $s_d^1(d = 1, 2, 3)$ of the three candidate assets under factor "Repeatability" is calculated using Equation (8.7) according to Equation (5.10).

$$s_1^1 = 0.335, s_2^1 = 0.454, s_3^1 = 0.211 \tag{8.7}$$

The specification efficiency score of the three candidate assets under other evaluators is calculated using the same approach. Table 8.4 lists each candidate asset's scores under different sub-evaluators.

Table 8.4: Specification efficiency score of each candidate asset under single evaluators

Sub-Evaluators	**KUKA KR 1000 titan**	**ABB IRB 6700-150**	**FANUC M-900iB/360**
Repeatability	0.335	0.454	0.211
Payload	0.211	0.454	0.335
Reachability	0.211	0.335	0.454

In this application, equal importance is assigned to each sub-evaluator, which means the weight of each sub-evaluator is 0.33. The total specification efficiency score (TSES) can be calculated in Equation (8.8):

$$\text{TSES} = \frac{1}{3} \times s_d^1 + \frac{1}{3} \times s_d^2 + \frac{1}{3} \times s_d^3 \tag{8.8}$$

Table 8.5 displays the total specification efficiency scores for each candidate asset. Based on the calculation result, the ABB IRB 6700-150 robot is selected as the candidate robot to execute the assembly task, as it has the highest total specification efficiency score.

Table 8.5: Total specification efficiency scores of each candidate asset

	KUKA KR 1000 titan	**ABB IRB 6700-150**	**FANUC M-900iB/360**
Total specification efficiency score	0.252	0.414	0.333

The layout optimisation semantic model also assists in this task by providing recommendations

for layout optimisation via ontology reasoning in the knowledge graph (experience databank). These recommendations pertain to the objectives, decision variables, constraints, and reconfiguration solutions in the asset model that should be considered during the reasoning process, as illustrated in Figure 8.14.

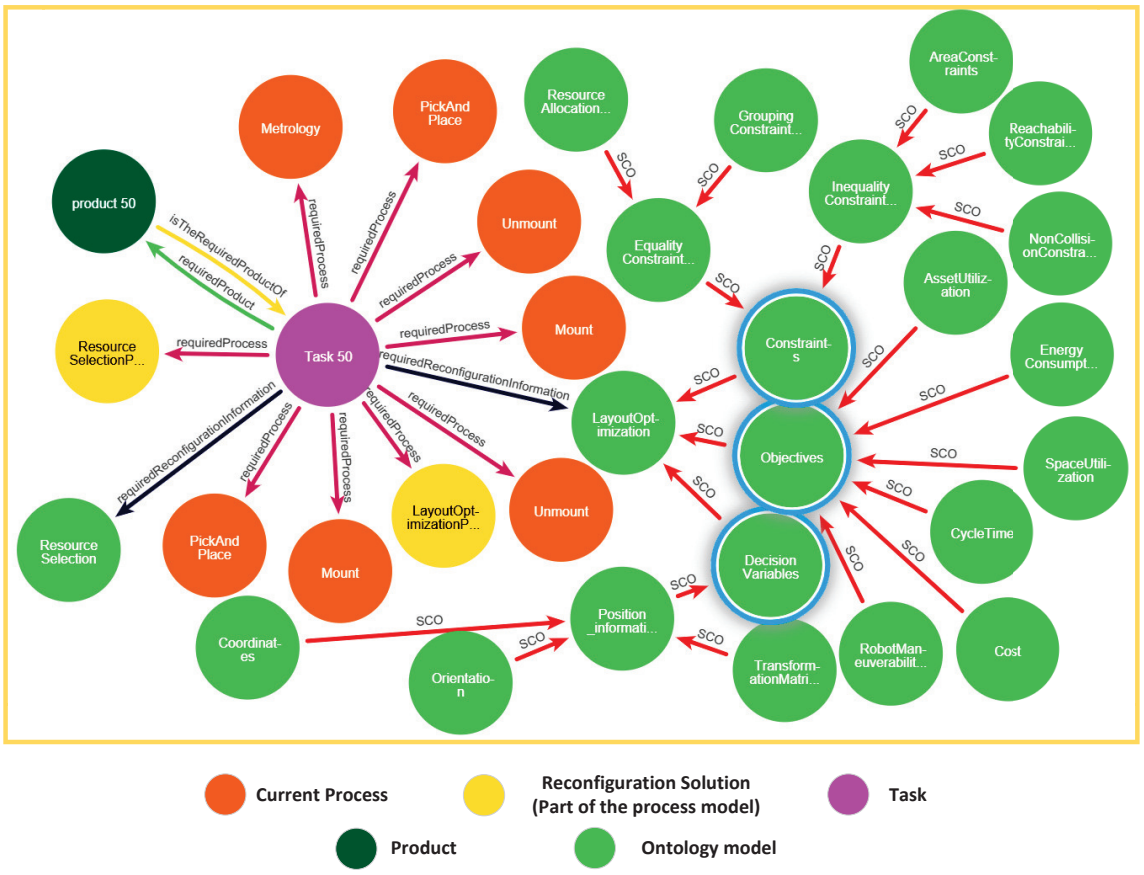


Figure 8.14: Reconfiguration model to enhance the layout reconfiguration process

The reconfiguration model acquires recommendations by querying the process model and capability model as shown in Figure 4.2. The capability model supplies information about potential assets capable of executing the reconfiguration solution process. For instance, the reconfiguration process necessitates resource selection and layout optimisation, as discussed in Task 50. The capability model offers potential assets to execute the "ProcessNeedsReconfiguration" and information regarding the reconfiguration solution required for resource selection. Moreover, the capability model determines whether it is essential to change the current assets for reconfiguration. The reconfiguration solution for layout optimisation is identified through the connection between the capability semantic model and the asset semantic model. As shown in Figure 8.15, various reconfiguration solutions are available to address the layout optimisation problem.

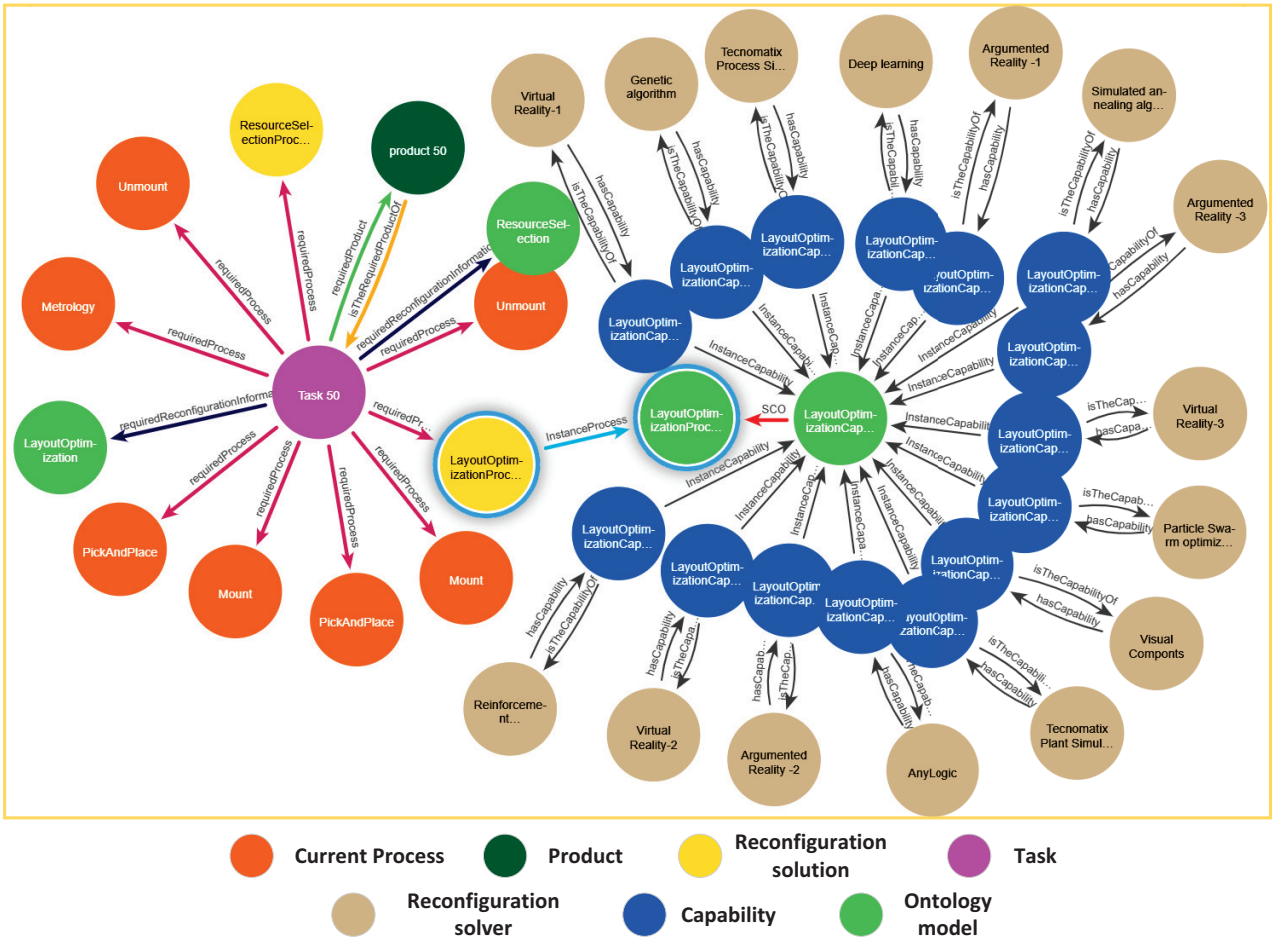


Figure 8.15: Recommendations from the knowledge graph to solve the layout optimisation problem

The optimisation process is executed based on the enhanced information from the reconfiguration, capability, capacity, and process models. The framework of the layout optimisation for this use case is pictured in Figure 8.16. The simulation environment is created based on the results from the asset selection (reconfiguration stage 1) and information from the real equipment. The simulation environment sends information to the optimisation environment through a socket based on the Tecnomatix .NET API. This API is connected to the Tecnomatix Process Simulate simulation environment with Tecnomatix .NET viewers.

In the given use case, the cycle time is selected as the "user-defined" KPI targeted for improvement. The layout optimisation framework contains a loop of the optimisation process in the optimisation environment. After one optimisation is completed, the updated robot parameter is sent to the simulation environment to obtain a new cycle time and then check for collisions. The Cyclic Event Evaluator (CEE) simulation mode in Tecnomatix Process Simulate is used in this example [69]. CEE, which functions as a PLC, controls how a typical robotics simulation

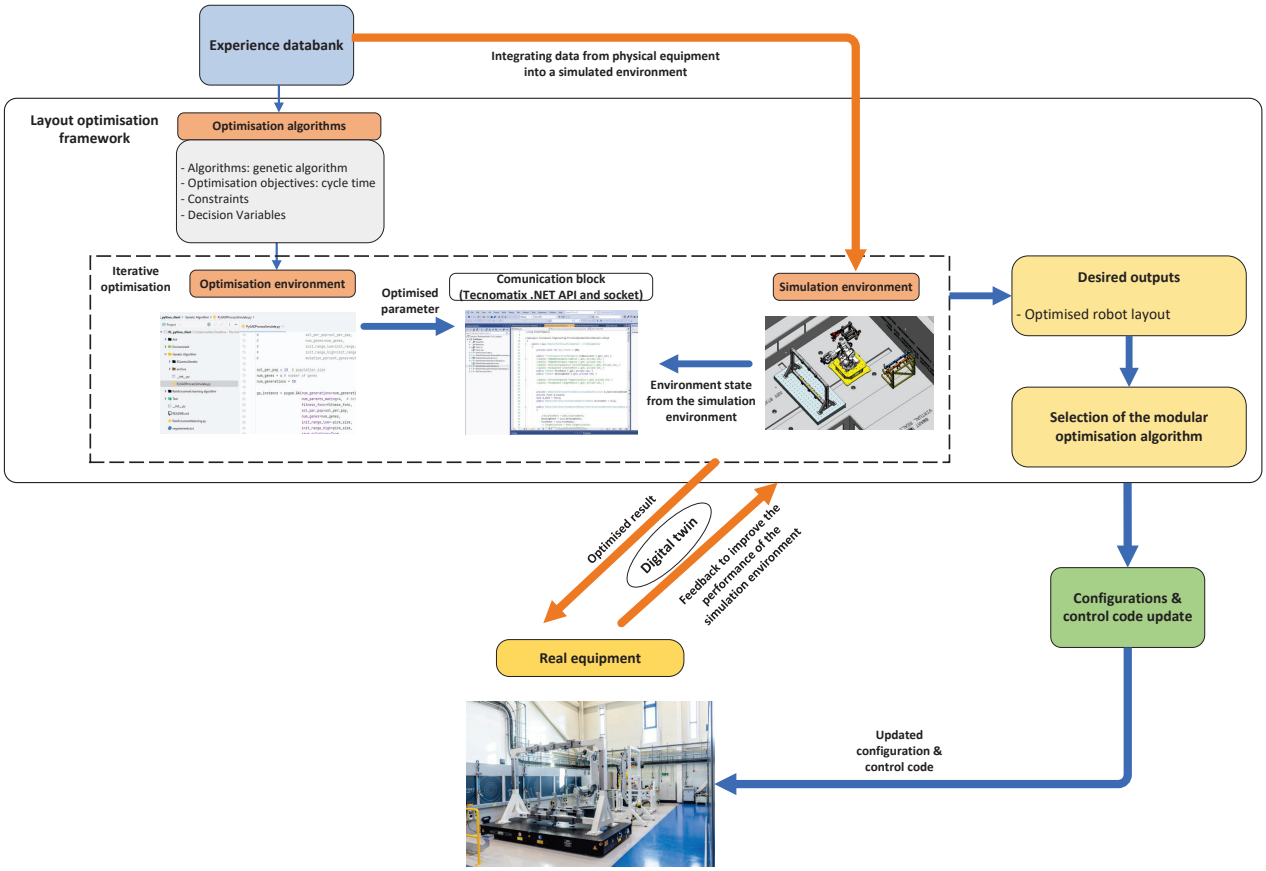


Figure 8.16: Utilisation of the layout optimisation framework in the Use Case 1

progresses using logic. Once the start signal is true, the simulation commences. Initially, no robot move relocation functions are defined. With the Tecnomatix .NET API, each simulation can generate a move operation. In the first iteration, the optimisation environment sends random coordinates of both robots to the simulation environment, and then two object flow operations for relocating the robots in Tecnomatix Process Simulate are generated and linked with other operations.

Figure 8.17 displays the initial layout of the current production cell. The required cycle time is 47.17 seconds; after optimisation with the genetic algorithm, the cycle time decreases to 39.83 seconds. In other words, the cycle time has been reduced by 15.6%. The corresponding optimised layout is shown in Figure 8.18, and the optimisation curve is depicted in Figure 8.19.

In summary, the reconfiguration model can perform tasks to optimise resource selection and layout optimisation processes:

1. Provide recommendations about the optimisation objectives, constraints, and decision variables to be considered in the reconfiguration optimisation problem.

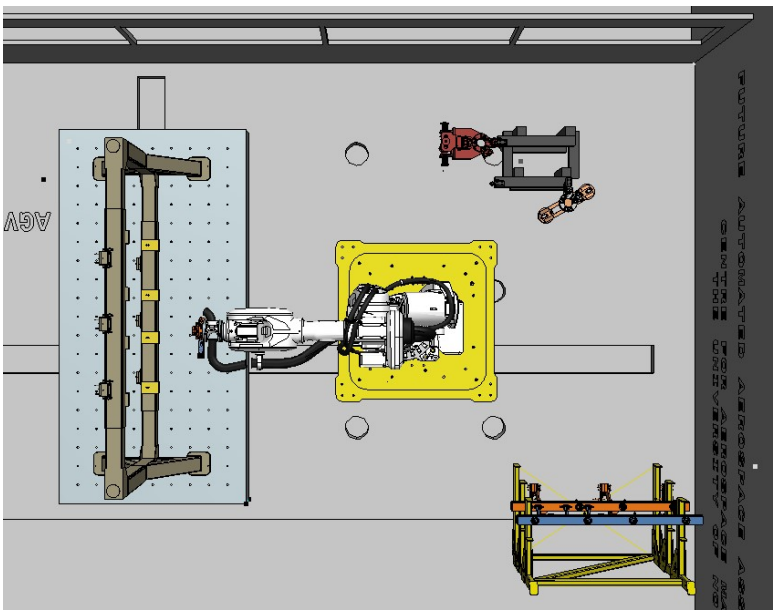


Figure 8.17: Initial work cell layout

2. Aid the reconfiguration-related task in identifying the potential assets to perform the reconfiguration task (in this case, resource selection and layout optimisation).

8.1.4 Discussion of the Results of Use Case 1

Section 8.1 presents a use case for employing the proposed experience databank for asset selection and provides recommendations for layout reconfiguration. The proposed methodology, which combines top-down and bottom-up strategies, streamlines the construction and updating of the knowledge graph, thereby offering significant advantages for intelligent search, tailored recommendations, and perceptive query resolution.

The capability of the knowledge graph to manage real-time inquiries and dynamic modifications is validated through two distinct applications. The first application focuses on a non-reconfiguration task, which employs the experience databank to identify the most appropriate assets and requisite processes for product manufacturing. Moreover, this application demonstrates the process of decomposing combined capabilities to uncover potential assets for the input capability of the combined capability. The second application examines a reconfiguration task from the OMNIFACTORY project at the University of Nottingham. It involves the application of the reconfiguration model, illustrating both current processes necessitating reconfiguration and reconfiguration solutions in specific reconfiguration types. In this use case,

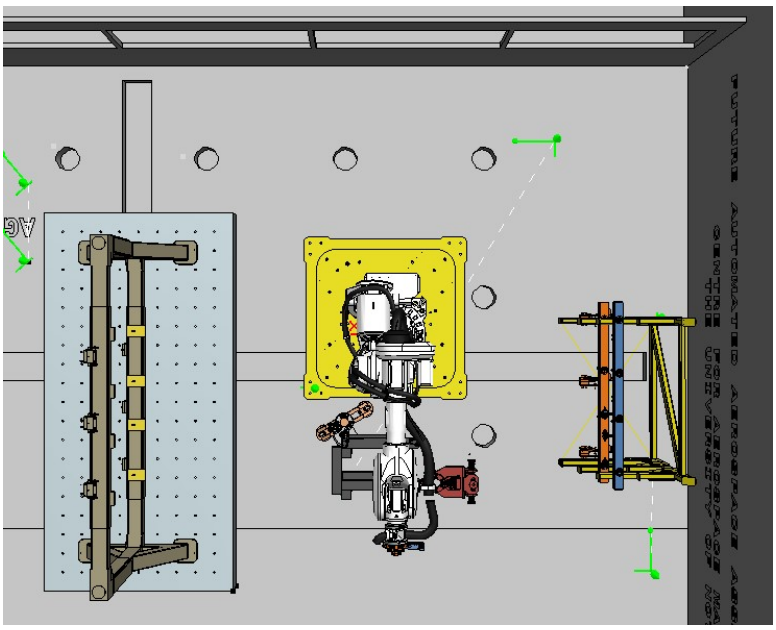


Figure 8.18: Optimised work cell layout

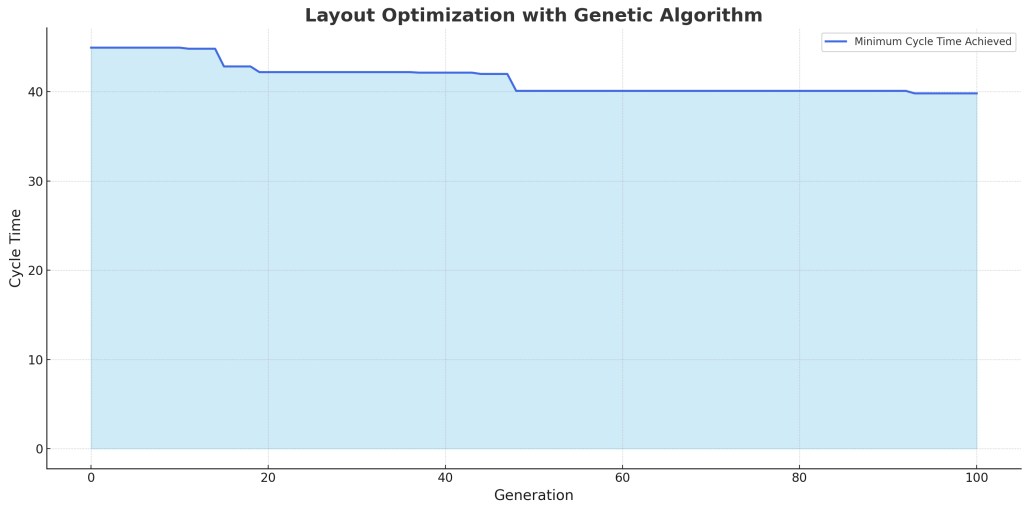


Figure 8.19: Optimisation fitness cure of the genetic algorithm

both asset selection and layout optimisation with the help of the experience databank are discussed. Asset selection was executed to find the most appropriate candidate assets based on the specification score, and layout optimisation was performed to identify the optimised layout based on the optimisation objective cycle time. The criteria in Table 8.6, with check marks for the objectives, are validated.

For Use Case 1, the validation was methodically executed against a set of predefined criteria to establish the robustness of the methodology. The details of the validation are summarised below:

Table 8.6: Validation according to the criteria in Use Case 1

			Validation
Contribution 1	Criterion 1.1	Vendor Neutrality	✓
	Criterion 1.2	Information Modelling	✓
	Criterion 1.3	Handling of Data from Diverse Sources	✓
	Criterion 1.4	Reasoning based on the Ontology	✓
Contribution 2	Criterion 2.1	Adapting to New Process Requirement	✓
	Criterion 2.2	Capability Assessment	✓
	Criterion 2.3	Modular Asset Selection	
	Criterion 2.4	Multi-Criteria Asset Selection	
	Criterion 2.5	Synergies with System	
Contribution 3	Criterion 3.1	Multi-Criteria Layout Optimisation	
	Criterion 3.2	Modular Layout Optimisation	
	Criterion 3.3	Interoperability	✓
	Criterion 3.4	Scalability	

- **Criterion 1.1 (Vendor Neutrality):** The OCCR ontology model effectively captures robot assembly cell data in a vendor-agnostic manner, resulting in the validation of this criterion in Use Case 1.

- **Criterion 1.2 (Information Modelling):** The use of the ontology model to encompass crucial robotic assembly reconfiguration data, such as capability, capacity, and reconfiguration details, validates this criterion.

- **Criterion 1.3 (Handling of Data from Diverse Sources):** The methodology successfully integrates diverse data, including data related to robots, end effectors, processes, and tasks, into the experience databank, thus validating this criterion.

- **Criterion 1.4 (Reasoning Based on Ontology):** Demonstrated by the capability of ontology reasoning to extract inferred knowledge, such as matching capabilities and decomposing processes. This validation is evident from its application in areas such as capability matching for marking tasks and decomposition processes for pick-and-place tasks.

- **Criterion 2.1 (Adapting to New Process Requirement):** The methodology's adaptability to new task requirements, such as tasks 50 and 100, underscores its ability to cater to new process requirements, thereby validating this criterion.

- **Criterion 2.2 (Capability Assessment):** In this use case, the examination of capabilities and the subsequent matching of specific capabilities, such as pick-and-place and marking capabilities, lead to the confirmation of this criterion.

- **Criterion 3.3 (Interoperability):** Creating a layout optimisation environment facil-

itated by seamless asset communication is a testament to the methodology's interoperability, which confirms this criterion is validated.

8.2 Use Case 2 – Multi-Criteria Layout Optimisation in a Single Robot via Experience Databank

In Use Case 1, several criteria were not satisfied, including Criterion 2.3 (Modular Asset Selection), Criterion 2.4 (Multi-Criteria Decision Making), Criterion 2.5 (Synergies with System), Criterion 3.1 (Multi-Criteria Layout Optimisation), Criterion 3.2 (Modular Layout Optimisation), and Criterion 3.4 (Scalability). This case focused primarily on layout optimisation at a system level with a singular objective: cycle time. To address these deficiencies, Use Case 2 was introduced. The focus of Use Case 2 is shown in Figure 8.20. Its main aim is to validate the criteria unmet in Use Case 1 and to refine the validation of others. Unlike the system-level emphasis of Use Case 1, Use Case 2 concentrates on the machine level. A notable feature is the optimisation of a single robot's pose using a multi-objective approach. However, asset selection is not discussed in Use Case 2; thus, Criterion 2.3 (Modular Asset Selection), Criterion 2.4 (Multi-Criteria Asset Selection), and Criterion 2.5 (Synergies with System) are not validated in Use Case 2 and are set to be addressed in Use Case 3.

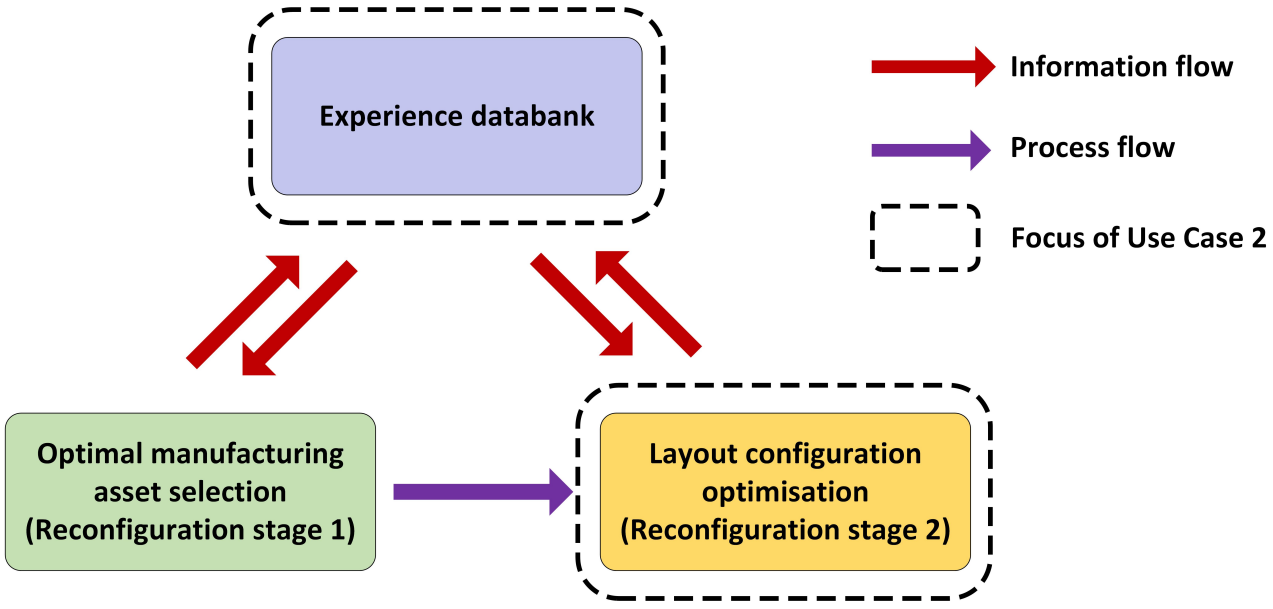


Figure 8.20: Validation components of the proposed methodology

8.2.1 Introduction

Optimising the layout of industrial robots is an integral facet of production line planning. This complex task involves the precise determination of the robot's position and orientation, allowing for the effective and efficient execution of designated tasks. An optimally laid-out robot ensures the desired position and orientation of its end effector while circumventing potential collisions, thereby aiding in achieving operational efficiency and energy conservation targets.

The understanding of a robot's working range, often termed the "working envelope" [120], is a fundamental step in robot operations. This workspace is primarily defined by the robot's external structure and physical dimensions, serving as a central point for optimisation in many industrial settings.

In this use case, a specific component of the methodology introduced in the PhD thesis is examined. The broader methodology entails various elements, but the focus here is on the centralisation of the experience databank and layout optimisation techniques, particularly for a single machine with multiple objectives.

The presented use case aims to validate this specific part of the methodology by determining the optimal position for a single robot pose during particular drilling operations. The scenario involves a workpiece with 30 fixed-position holes set to be drilled. The goal is to pinpoint the most suitable placement of the workpiece to ensure efficient robot operations, taking into account multiple objectives. The drilling depth is established at 5 mm. Potential robot poses, and positions are displayed in Figure 8.21, and Figure 8.22 offers a view of the workpiece.

Upon validation, this use case has extensive potential for real-world applications in various industries. By simply altering the operation points for new tasks, the rest of the procedure remains consistent, thereby demonstrating the versatility and adaptability of the proposed methodology. The workflow of applying this use case is depicted in Figure 8.23.

8.2.2 Establishment of the Experience Databank

FANUC ER-4iA is an example of a typical six-axis industrial robot that could be used for drilling applications. Based on information about the drilling hole, the appropriate end effector

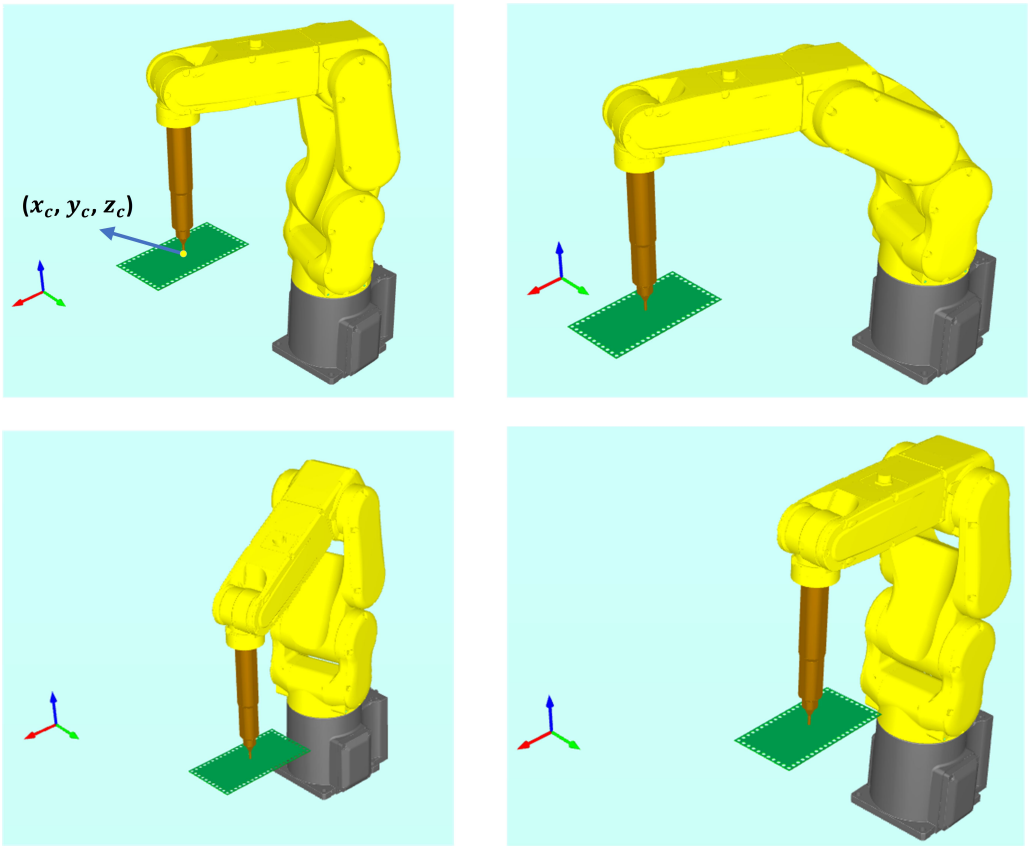


Figure 8.21: Possible poses of doing the drilling operation

is selected.

The experience databank also helps with the layout reconfiguration process; it suggests the optimisation objectives that should be used and the reconfiguration solver as described in Section 4.2.2.9. Energy consumption, robot manoeuvrability, and cycle time optimisation are suggested as the optimisation objectives in this use case. As this use case involves layout optimisation of a single robot with a single operation (drilling), RoboDK is the simulation environment of choice.

In summary, the inception of the experience databank constitutes a vital initial stage in the prescribed methodology. These tools augment the optimisation of robotic layouts while aligning with the broader goals of operational efficiency and energy conservation. Their adaptability to various robots showcases the versatility and scalability inherent in the proposed methodology.

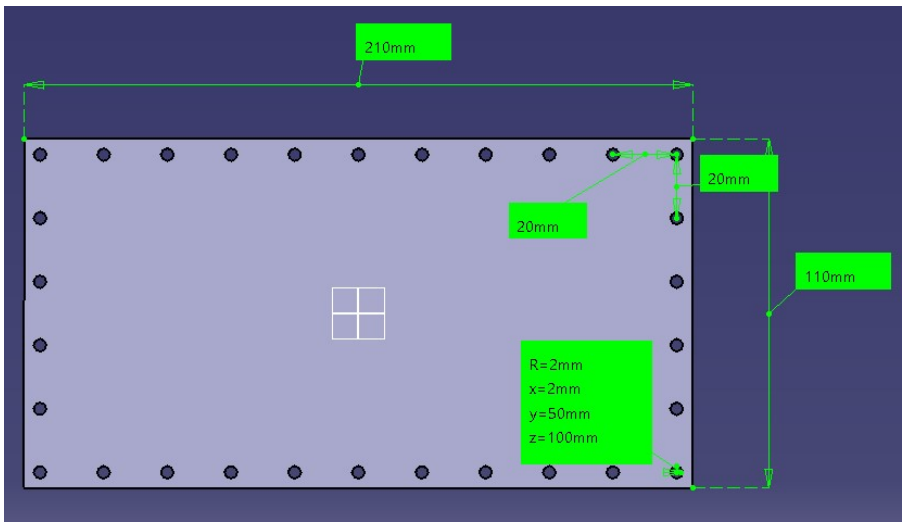


Figure 8.22: Workpiece information

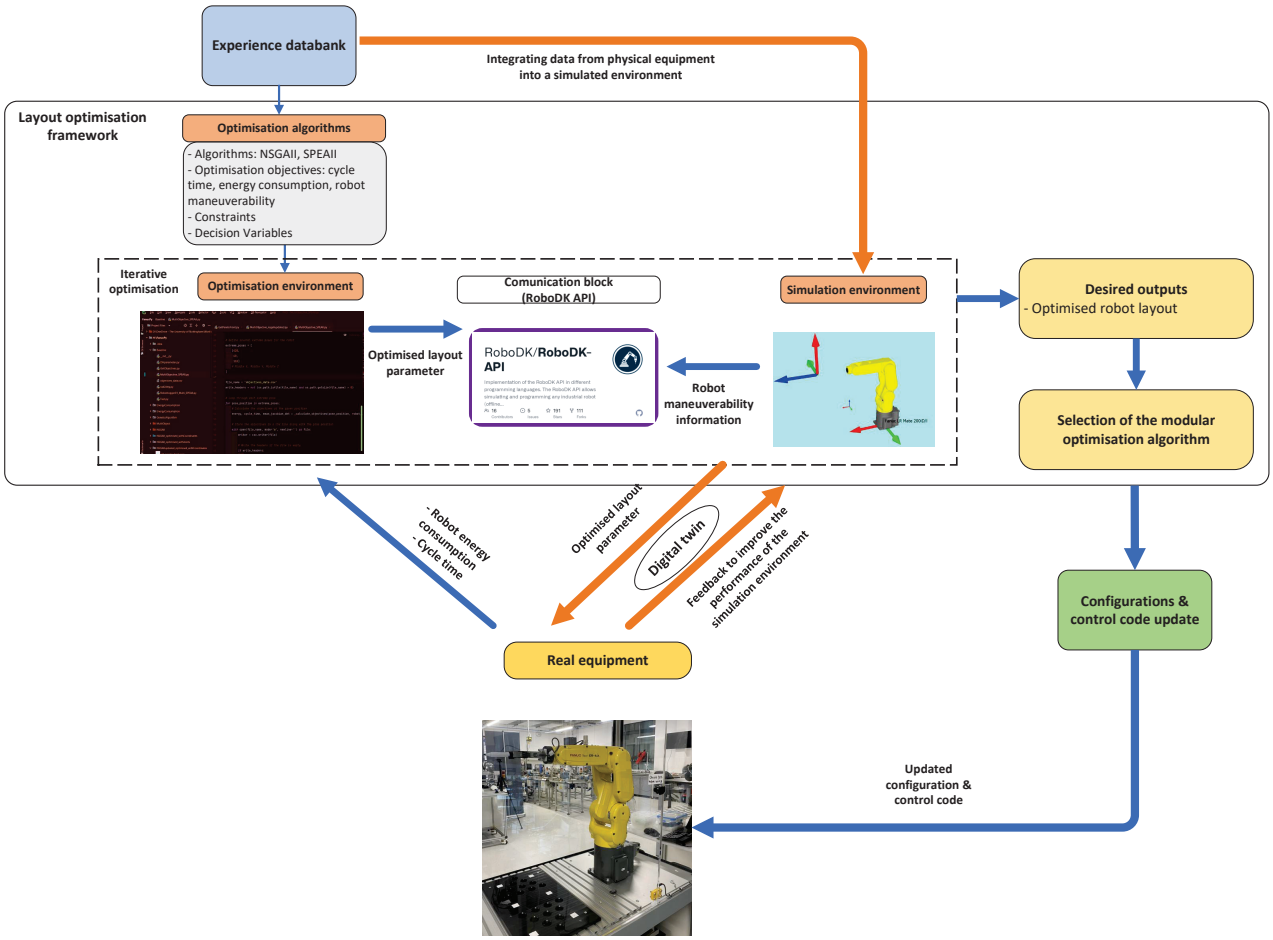


Figure 8.23: Workflow of the layout optimisation in Use Case 2

8.2.3 Layout Configuration Optimisation

Having utilised the experience databank information for this task, the system recommends a reference for optimising the layout. The experience databank provides detailed insights for this

use case, specifying the necessary robots, the required DH parameters for the robot, and other related information crucial for layout optimisation.

8.2.3.1 Simulation Environment

The aim of this particular use case is to identify an optimal pose for a single robot engaged in a singular operation. RoboDK excels in this area and is, therefore, the software chosen for the simulation environment in this instance.

To affirm the applicability of the suggested framework across a range of scenarios, RoboDK has been utilised as a digital twin environment, as depicted in Figure 8.24. Some of the many adaptable features of RoboDK include its ability to offer robot controllers and kinematics solvers customised to a variety of robot brands. In this instance, the internal solver of this digital twin environment has been employed to calculate the end effector pose and to verify the reachability of all drilling points based on the DH parameters.

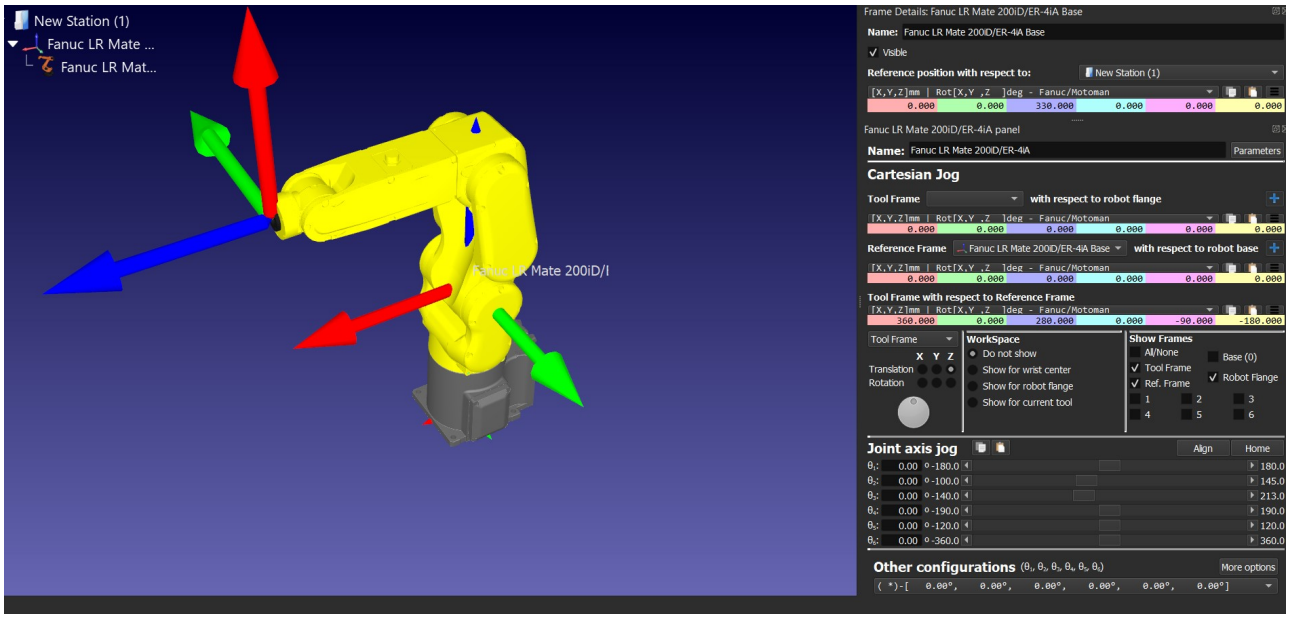


Figure 8.24: Interface for the RoboDK

8.2.3.2 Optimisation Environment

The environment, in this case, is deployed locally, and the code is written in Python. The optimisation environment is connected with the simulation environment with the RoboDK API and with the real robot with FANUCPy [121] via Socket.

8.2.3.3 Optimisation Algorithm

Given the multi-objective nature of the problem, the NSGA-II, SPEA-II and Non-Dominated Sorting Genetic Algorithm III (NSGA-III) algorithms are chosen for their suitability in addressing such challenges. Several reasons underpin this selection:

1. **Proficiency in Multi-Objective Problems**: NSGA-II, SPEA-II and NSGA-III are renowned for their capability to handle multi-objective optimisation problems. They excel in generating a diverse set of Pareto-optimal solutions, ensuring that the solution space is explored comprehensively.
2. **Integration in Experience Databank**: A significant advantage of the three algorithms is their pre-storage within the experience databank. Using pre-stored algorithms in experience databanks not only offers computational efficiency but also underscores reliability, given their prior validation and optimisation for analogous tasks.

The parameters pivotal for layout optimisation, such as decision variables, constraints and optimisation objectives, are detailed in Sections 8.2.3.3.1, 8.2.3.3.2 and 8.2.3.3.3.

8.2.3.3.1 Decision Variables In this use case, the decision variables correspond to the coordinates of the workpiece's centre. These coordinates provide the basis for determining each drilling point. Consequently, the robot's end effector pose, or the Tool Center Point (TCP), is influenced during drilling operations. The pose of the end effector is a function of the robot's joint angles, which are inherently tied to the DH parameters.

DH parameters offer a standardised method to describe the kinematic structure of robotic arms. They encapsulate the spatial relationship between adjacent robot links, simplifying the robot's mathematical representation. By employing these parameters, a clear connection between the decision variables (coordinates of the TCP) and the robot's joint configurations emerges. This relationship becomes vital for assessing feasible robot movements and achieving desired TCP positions.

Using inverse kinematics and the end effector pose at each drilling point, whether the robot can effectively reach all drilling points based on the given decision variables can be ascertained.

The FANUC ER-4iA robot's default DH parameters are outlined in Table 8.7. However, in real-world drilling scenarios where a drilling gun is integrated into the robot, the DH parameters need adjustment, notably to account for the drilling tool's 250 mm length. The adapted parameters are listed in Table 8.8.

Table 8.7: Modified Denavit-Hartenberg parameters of FANUC ER-4iA

link$_i$	α_{i-1} (°)	a_{i-1} (mm)	d_i (mm)	θ_i (°)	joint limits (°)
1	0	0	330	θ_1	[-170, 170]
2	-90	0	0	θ_2 - 90	[-110, 145]
3	0	260	0	θ_3	[-122, 280]
4	-90	20	290	θ_4	[-190, 190]
5	90	0	0	θ_5	[-120, 120]
6	-90	0	70	θ_6	[-360, 360]

Table 8.8: Modified Denavit-Hartenberg parameters of FANUC ER-4iA with considering the length of the end effector

link$_i$	α_{i-1} (°)	a_{i-1} (mm)	d_i (mm)	θ_i (°)	joint limits (°)
1	0	0	330	θ_1	[-170, 170]
2	-90	0	0	θ_2 - 90	[-110, 145]
3	0	260	0	θ_3	[-122, 280]
4	-90	20	290	θ_4	[-190, 190]
5	90	0	0	θ_5	[-120, 120]
6	-90	0	70 + 250 = 320	θ_6	[-360, 360]

The current use case seeks an operation point that is energy efficient, ensures optimal robot operability, and aims to reduce the cycle time. Potential robot poses aligned with this objective can be seen in Figure 8.21. As emphasised earlier, the decision variables, defined by the coordinates x_c, y_c and z_c, pinpoint the centre of the workpiece. With this centre point established, the coordinates of the operation points can be calculated.

8.2.3.3.2 Constraints Various constraints must be taken into account in the optimisation process, such as joint limits, collision avoidance, and specific task requirements. These constraints are crucial to ensure the operational feasibility and durability of robotic equipment.

Joint limits represent inherent constraints emanating from the robot's design. They stipulate the maximum permissible range of motion for each joint to avoid causing damage. These limits must be incorporated into the optimisation process to maintain the safety and functionality of the robot.

The inclusion of collision avoidance as a key constraint is also of paramount importance. Robots must be programmed to prevent collisions with each other or other objects in the workspace.

This helps to maintain the integrity of the workflow, prevent damage and ensure a safe working environment.

There might be other constraints related to workspace limitations, a robot's operational specifications, or specific task requirements. These constraints, decision variables and optimisation objectives must be carefully balanced to achieve an optimal layout configuration.

The operation of industrial robots such as FANUC ER-4iA, especially in educational settings like the FANUC education cell, necessitates a strong emphasis on safety. Establishing a safe operating space for the robot ensures operator safety and the protection of surrounding equipment. As illustrated in Figure 8.25, a predefined boundary for the robot's end effector is demarcated. This boundary constrains the range of the end effector, mitigating potential risks and preventing unintentional interactions with external objects or personnel. A defined safe working space ensures a secure environment while enabling the effective performance of the FANUC ER-4iA robot.

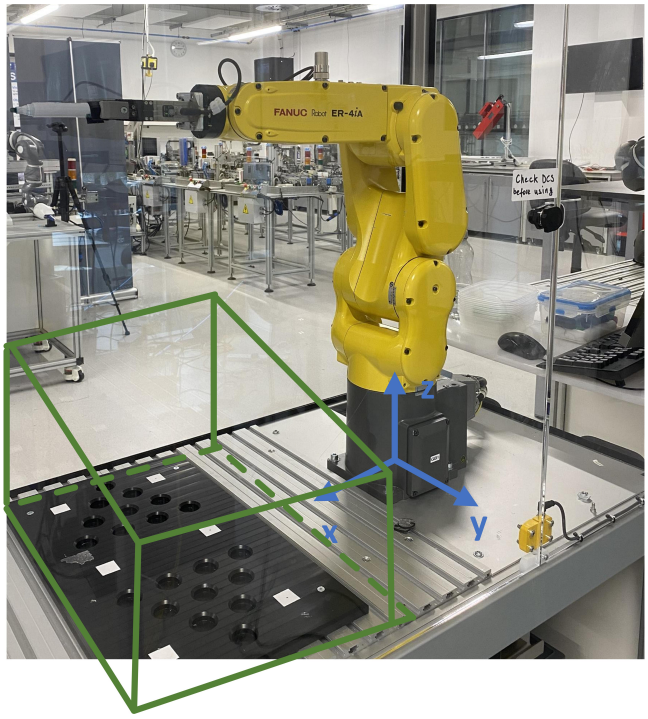


Figure 8.25: Relationship between robot and workspace

A safety range is specified for each axis: x, y, and z. These ranges ensure that objects within this space adhere to certain constraints or operate within a specific area. The specific safety

ranges for each axis are as follows:

$$x \in [300, 500], \quad y \in [-150, 150], \quad z \in [50, 350]$$

For convenience, it is assumed that the orientation of the end effector pose is (-180, 0, 0) as depicted in Figure 8.26. Thus, the end effector pose in this case is (x_c, y_c, z_c, -180, 0, 0) to represent the centre point of the workpiece position.

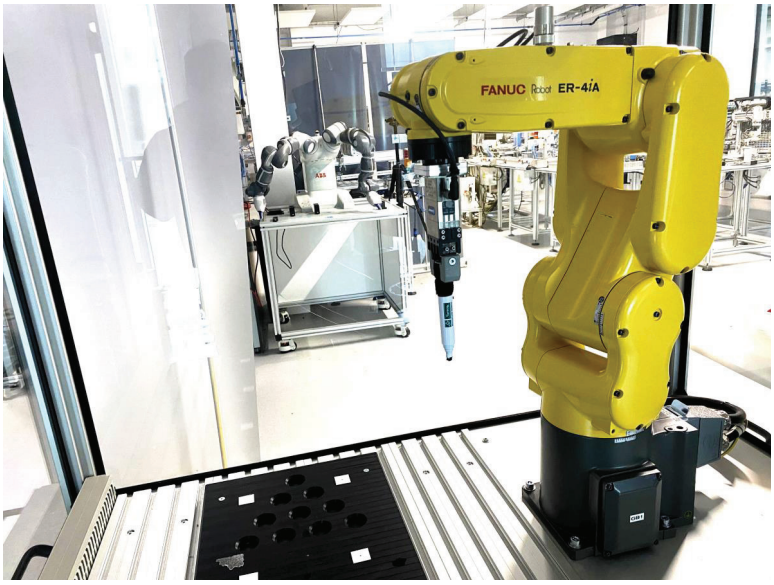


Figure 8.26: Orientation of the end effector pose

8.2.3.3.3 Optimisation Objectives The primary optimisation objectives in this scenario are energy consumption, robot manoeuvrability and cycle time.

8.2.3.3.3.1 Energy Consumption Consideration of energy consumption is vital, particularly in light of sustainable manufacturing practices. A reduction in energy usage diminishes operational costs and promotes environmental sustainability. The optimisation of joint angles, therefore, aims to minimise the energy consumed by the robot during its operation.

The energy consumption of the robot is recorded using a robot controller and a robot logger generated on FANUCPy [121], which makes the robot move and record energy consumption correspondingly. The method of acquiring real-time energy consumption readings involves a distinct series of steps, as depicted in Figure 8.27.

The initial stage of this process involves the assignment of input decision variables facilitated

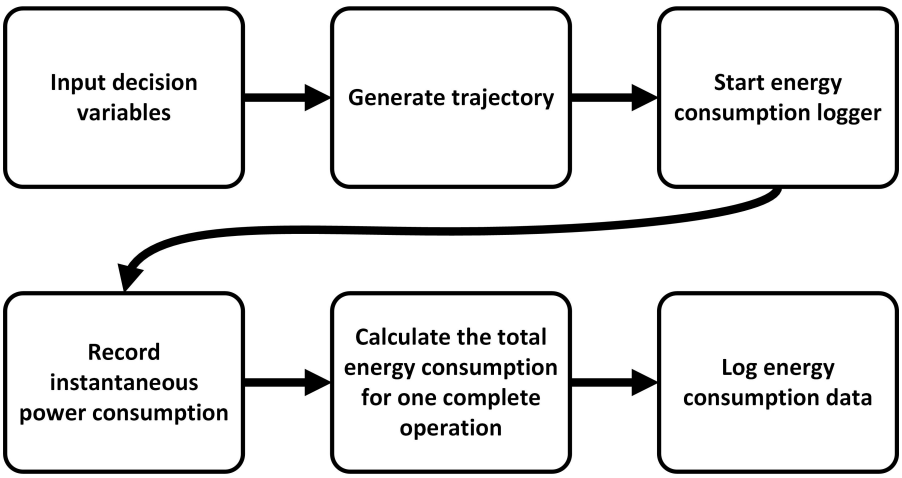


Figure 8.27: Flowchart of recording energy consumption

through a sampling method. Subsequently, a trajectory corresponding to these variable values is generated. The initial phase concludes with the activation of the energy consumption logger, which triggers the recording of energy consumption.

Thereafter, the robot controller assumes control of the robot's actions, which operate based on commands relayed from a Python script. This process stage is of utmost importance, encompassing the logging and recording of instantaneous power consumption. This data is then relayed from the robot back to the Python script, enabling accurate, real-time recording of power usage during operation.

In this use case, the recording frequency is set at 50 Hz. This frequency allows for data recording corresponding to a complete operational cycle, during which the robot navigates to all the designated drilling points.

Upon completion of one operational cycle, the total energy consumption is calculated based on the duration of the cycle and the accumulated instantaneous power consumption. The process concludes with the computation of the overall energy consumption. This systematic approach allows for detailed energy consumption profiles to be captured during robotic operations, thereby informing improvements in energy efficiency and operational effectiveness.

8.2.3.3.3.2 Robot Manoeuvrability A robot's manoeuvrability refers to its efficiency in movement and positioning within its operational space. Such enhanced capability ensures adaptability to changes in the environment or tasks, leading to improved efficiency and flexibility in the manufacturing process. This concept has been detailed in Section 6.2.3.6.2.

In this use case, it is assumed that the robot must navigate to a total of n points. The end effector pose at the drilling points can be described as per Equation (8.9).

$$\forall a \in 1, 2, ..., n, P_a = (x_a, y_a, z_a, -180, 0, 0) \tag{8.9}$$

where P_a denotes the coordinates of the end effector pose at the corresponding operational point.

According to inverse kinematics, the joint angles of each point can be calculated as follows:

$$\forall a \in \{1, 2, ..., n\}, \theta_a = IK(P_a) \tag{8.10}$$

where θ_a is the joint vector for point a.

$$P_a = f(\boldsymbol{\theta_a}) = \begin{pmatrix} f_1(\boldsymbol{\theta_a}) \\ f_2(\boldsymbol{\theta_a}) \\ f_3(\boldsymbol{\theta_a}) \\ -180 \\ 0 \\ 0 \end{pmatrix} \tag{8.11}$$

Here, P_a denotes the end effector pose at the operation point a, which is a function of the joint angle vector θ_a. The function f denotes the forward kinematics of the robot, mapping the joint angles to the Cartesian coordinates x_a, y_a, z_a. The Euler angles remain constant at $(-180, 0, 0)$ for each operation point as per the given information.

For the computation of robot manoeuvrability in the specific case of n drilling points, the same principles as described above apply. The differentiation of the end effector pose function, as presented in Equation (8.11), with respect to the joint angle vector, results in the Jacobian matrix of the system:

$$\boldsymbol{J}(\boldsymbol{\theta_a}) = \begin{pmatrix} \frac{\partial f_1}{\partial \theta_{a1}} & \frac{\partial f_1}{\partial \theta_{a2}} & \frac{\partial f_1}{\partial \theta_{a3}} & \frac{\partial f_1}{\partial \theta_{a4}} & \frac{\partial f_1}{\partial \theta_{a5}} & \frac{\partial f_1}{\partial \theta_{a6}} \\ \frac{\partial f_2}{\partial \theta_{a1}} & \frac{\partial f_2}{\partial \theta_{a2}} & \frac{\partial f_2}{\partial \theta_{a3}} & \frac{\partial f_2}{\partial \theta_{a4}} & \frac{\partial f_2}{\partial \theta_{a5}} & \frac{\partial f_2}{\partial \theta_{a6}} \\ \frac{\partial f_3}{\partial \theta_{a1}} & \frac{\partial f_3}{\partial \theta_{a2}} & \frac{\partial f_3}{\partial \theta_{a3}} & \frac{\partial f_3}{\partial \theta_{a4}} & \frac{\partial f_3}{\partial \theta_{a5}} & \frac{\partial f_3}{\partial \theta_{a6}} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{8.12}$$

where θ_{ai} represents the joint angle of the i^{th} joint of the robot, for $i = 1, 2, 3, 4, 5, 6$.

The determinant of the Jacobian matrix $\boldsymbol{J}(\boldsymbol{\theta_a})$ is ascertained for each operation point a. For cases where the determinant is non-zero, its absolute value is computed using the formula given

by Equation (6.15) and is denoted as $w_a = |\det \boldsymbol{J}(\boldsymbol{\theta_a})|$.

If the determinant of the Jacobian matrix is zero or near zero, it indicates that the robot loses one degree of freedom at the operation point a. This situation is unfavourable and should be avoided.

To calculate the total robot manoeuvrability over all n operation points, the individual manoeuvrability values w_a are summed over all n operation points, as given by the following equation:

$$W_{total} = \sum_{a=1}^{n} w_a \tag{8.13}$$

where W_{total} represents the total robot manoeuvrability for the entire task, which consists of reaching all n drilling points.

In this specific case, the robot is tasked with reaching n operation points. As such, the average manoeuvrability of the robot over these n points is calculated. This calculation is performed by first summing the individual manoeuvrability values, w_a, obtained from each operation point a, and then dividing the result by the total number of operation points n. The calculation can be formally represented as follows:

$$\overline{W} = \frac{1}{n} \sum_{a=1}^{n} w_a \tag{8.14}$$

where $\overline{W}$ denotes the average robot manoeuvrability for the entire task, which involves reaching all n operation points.

The calculation ensures adequate representation of the robot's ability to move and position itself effectively, taking into consideration all the operation points in the workspace. The larger $\overline{W}$ is, the more manoeuvrability the robot has. The scaling factor for the absolute value of the determinants of the Jacobian matrix in this use case is set as 1×10^7. With this scaling factor, the scaled robot manoeuvrability index in this use case is calculated.

8.2.3.3.3.3 Cycle Time As mentioned in Equation (6.3), the cycle time comprises the movement time, operation time, and wait time. In the current use case, there is only one robot and one operation; thus, the wait time is null. Therefore, in this scenario, the movement time and operation time must be calculated.

The movement time is determined by the robot's traversal across all operation points, while the operation time pertains to the drilling operation at each point. Two methods are available for recording time, as shown in Section 6.2.3.6.1. In this instance, a timer in the optimisation code is used to record the cycle time accurately in real time.

8.2.3.4 Evaluation

Layout optimisation ensues after defining the decision variables, optimisation objectives, and constraints extrapolated from the experience databank. The process parameters, such as the precision of the hole and the robot's rigidity during the drilling operation, are not included in this optimisation. It considers the appropriate positioning of the robot's end effector during the drilling process. Inverse geometry assists in determining the joint angles corresponding to various drilling points. This optimisation procedure is illustrated in Figure 8.28.

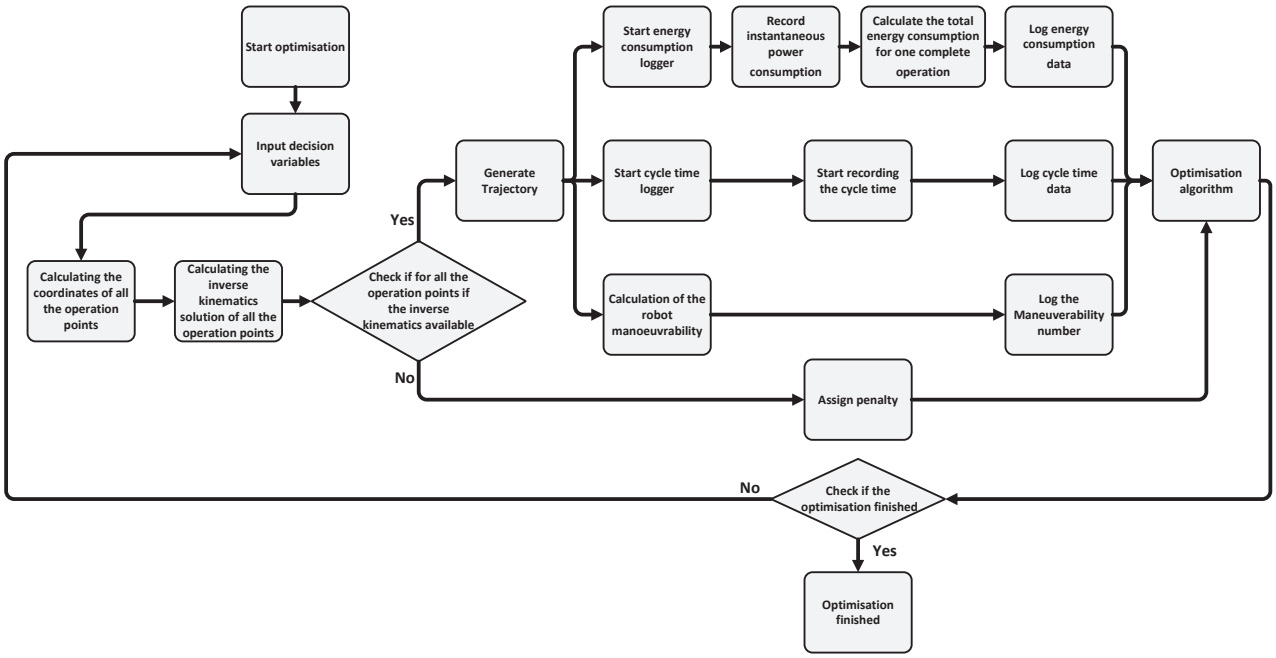


Figure 8.28: Flowchart of optimisation

The NSGA-II, SPEA-II and NSGA-III are proposed to solve the problem, with a generation number set to 100. As a baseline, the coordinates of the centre point of the workpiece should be (x_c: 400, y_c: 0, z_c: 200) as these coordinates fall in the middle of the value range. The baseline pose can be seen in Figure 8.29.

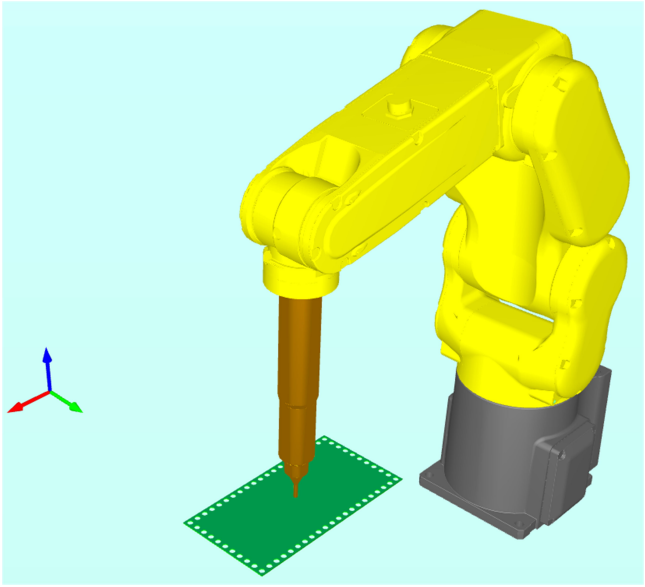


Figure 8.29: Baseline pose

When the workpiece is at this position, the energy consumption of the total drilling operation is 6,164.98 Wh, the cycle time is 57.11 seconds, and the scaled average determinant of the Jacobian matrix (Robot manoeuvrability index) is 0.4891235.

After the training of the multi-objective algorithms, the Pareto solutions can be identified. Pareto solutions obtained from NSGA-II are illustrated in Figure 8.30 and detailed in Table 8.9, where seven Pareto solutions are identified using NSGA-II. The following Pareto solutions are important for identifying the minimised or maximised optimisation objectives:

1. $S_{\text{NSGAII\_minE}}$ denotes the solution with the minimised energy consumption of 5, 613.59 Wh. The coordinates of the workpiece centre point for this solution are (x_c: 302.96, y_c: 76.83, z_c: 303.63).

2. $S_{\text{NSGAII\_minCT}}$ corresponds to the solution with the minimised cycle time of 53.15 s. The coordinates of the workpiece centre point for this solution are (x_c: 300.34, y_c: 87.62, z_c: 303.63).

3. $S_{\text{NSGAII\_maxRM}}$ denotes the solution achieving the maximised scaled robot manoeuvrability index of 1.1709163 (after scaling with 10^7). The coordinates of the workpiece centre point associated with this solution are (x_c: 303.33, y_c: 23.64, z_c: 260.10).

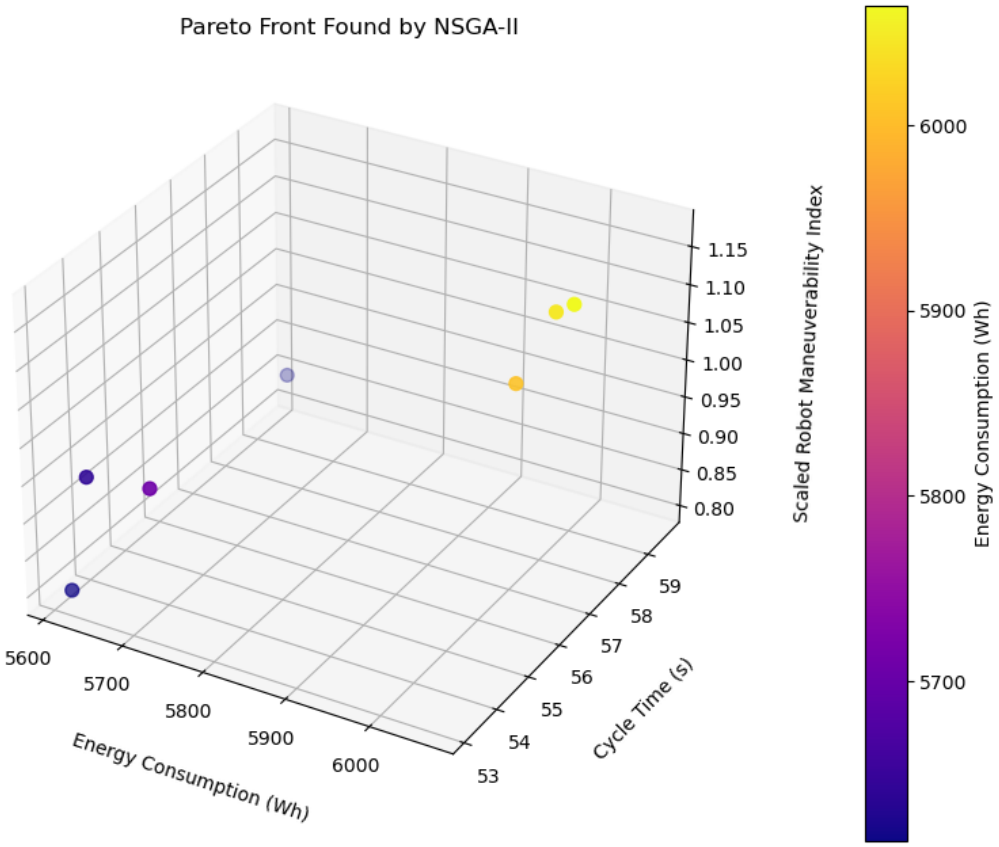


Figure 8.30: Pareto solutions obtained via NSGA-II

Table 8.9: Pareto solutions found by NSGA-II

Coordinates of the workpiece centre point	**Energy consumption (Wh)**	**Cycle time (s)**	**Scaled robot manoeuvrability index (scaled by 10^7)**
(x_c: 303.33, y_c: 23.64, z_c: 260.10)	6,064.71	57.11	1.1709163
(x_c: 300.34, y_c: 87.62, z_c: 303.63)	5,616.36	53.15	0.8033331
(x_c: 302.96, y_c: 76.83, z_c: 303.63)	5,613.59	59.64	0.8483343
(x_c: 300.12, y_c: 87.62, z_c: 260.42)	5,709.41	53.36	0.9626269
(x_c: 305.18, y_c: 15.17, z_c: 260.08)	6,006.82	56.90	1.0591016
(x_c: 300.01, y_c: 87.62, z_c: 260.30)	5,638.30	53.19	0.9610778
(x_c: 302.95, y_c: 23.64, z_c: 260.34)	6,045.36	57.07	1.1566736

Likewise, Pareto solutions obtained from SPEA-II are illustrated in Figure 8.31 and described in Table 8.10. From this table and figure, it is evident that five Pareto solutions are derived via SPEA-II. Identifying the following Pareto solutions is crucial to determine the minimised or maximised optimisation objectives:

1. $S_{\text{SPEAII\_minE}}$ denotes the solution that minimises energy consumption to 5,691.68 Wh. The coordinates of the workpiece centre point for this solution are (x_c: 303.73, y_c: 111.89, z_c: 247.91).

2. $S_{\text{SPEAII\_minCT}}$ corresponds to the solution with the minimised cycle time of 53.62 s. The coordinates of the workpiece centre point for this solution are (x_c: 303.73, y_c: 111.89, z_c: 256.25).

3. $S_{\text{SPEAII\_maxRM}}$ denotes the solution achieving the maximised scaled robot manoeuvrability index of 1.1662671. The coordinates of the workpiece centre point associated with this solution are (x_c: 334.87, y_c: 16.47, z_c: 283.90).

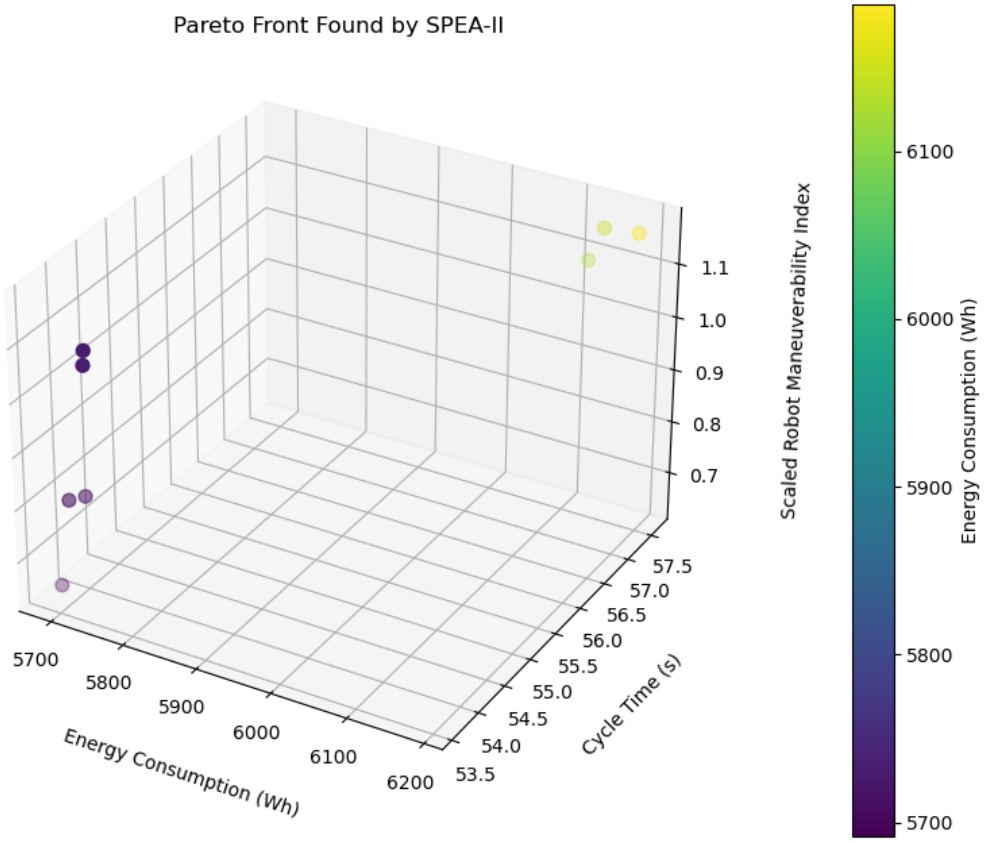


Figure 8.31: Pareto solutions obtained via SPEA-II

Table 8.10: Pareto front found by SPEA-II

Coordinates of the workpiece centre point	**Energy consumption (Wh)**	**Cycle time (s)**	**Scaled robot manoeuvrability index (scaled by 10^7)**
(x_c: 334.87, y_c: 16.47, z_c: 283.90)	6,187.44	57.57	1.1662671
(x_c: 300.73, y_c: 111.89, z_c: 256.25)	5,707.57	53.62	0.8174252
(x_c: 303.61, y_c: 111.89, z_c: 271.02)	5,731.46	53.65	1.0751479
(x_c: 302.71, y_c: 111.89, z_c: 271.02)	5,727.05	53.71	1.0946975
(x_c: 300.71, y_c: 111.89, z_c: 256.25)	5,719.44	53.75	0.8176363
(x_c: 300.73, y_c: 111.89, z_c: 247.91)	5,691.68	53.63	1.0973673
(x_c: 335.87, y_c: 13.42, z_c: 283.90)	6,128.13	57.48	1.1021526
(x_c: 334.87, y_c: 13.40, z_c: 283.90)	6,145.53	57.51	1.0973673

Subsequent to the multi-objective optimisation training, the Pareto solutions derived from NSGA-III are depicted in Figure 8.32 and detailed in Table 8.11. The table and figure indi-

cate that six distinct Pareto solutions are identified via NSGA-III. The key Pareto solutions highlighting the extremities of the optimisation objectives are as follows:

1. $S_{\text{NSGAIII\_minE}}$ denotes the solution with the minimised energy consumption: 5, 707.75 Wh. The coordinates of the workpiece centre point for this solution are (x_c: 312.18, y_c: 97.70, z_c: 238.78).

2. $S_{\text{NSGAIII\_minCT}}$ corresponds to the solution with the minimised cycle time of 53.67 s. The coordinates of the workpiece centre point for this solution are (x_c: 312.18, y_c: 97.70, z_c: 238.78).

3. $S_{\text{NSGAIII\_maxRM}}$ denotes the solution achieving the maximised robot manoeuvrability index of 1.1786125. The coordinates of the workpiece centre point associated with this solution are (x_c: 324.47, y_c: -0.12, z_c: 237.42).

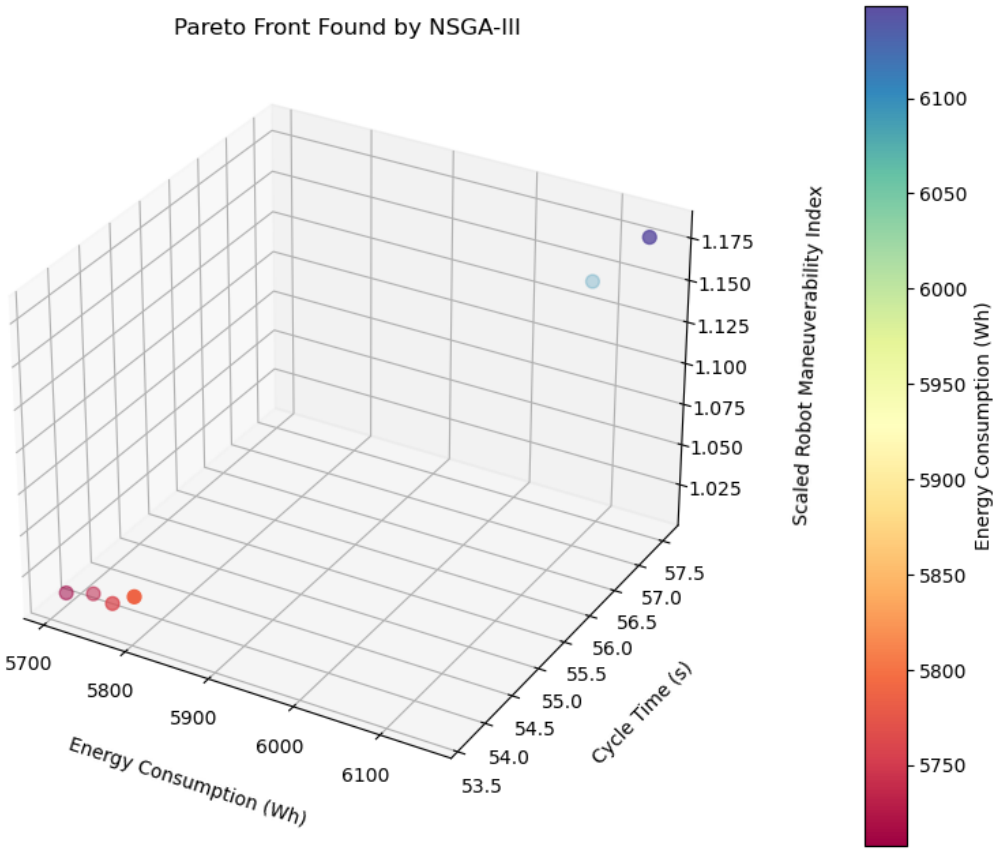


Figure 8.32: Pareto solutions obtained via NSGA-III

From comparisons of the three multi-objective algorithms – NSGA-II, SPEA-II, and NSGA-III – to optimise robotic functions, several observations can be made:

Table 8.11: Pareto front found by NSGA-III

Coordinates of the workpiece centre point	**Energy consumption (Wh)**	**Cycle time (s)**	**Scaled robot manoeuvrability index (scaled by 10^7)**
(x_c: 312.18, y_c: 97.70, z_c: 238.78)	5,707.75	53.67	1.0114668
(x_c: 312.24, y_c: 97.70, z_c: 238.78)	5,733.06	53.78	1.0115362
(x_c: 315.93, y_c: 97.70, z_c: 238.78)	5,788.35	53.72	1.0205584
(x_c: 324.46, y_c: -5.35, z_c: 237.60)	6,092.05	57.41	1.1484128
(x_c: 324.47, y_c: -0.12, z_c: 237.42)	6,148.32	57.54	1.1786125
(x_c: 312.24, y_c: 97.70, z_c: 238.78)	5,760.37	53.73	1.0115374

1. **Energy Consumption**:
    - NSGA-II provides the best results with the lowest energy consumption at $5,613.59$ Wh. This makes NSGA-II an ideal choice when optimising for energy efficiency.
    - SPEA-II and NSGA-III, with $5,691.68$ Wh and $5,707.75$ Wh respectively, have higher energy consumption. Neither exhibits a prominent advantage over the other with regard to this metric.
2. **Cycle Time**:
    - NSGA-II offers the best results in terms of cycle time, achieving a minimum of 53.15 s. Operations requiring rapid task completion might benefit most from the solutions provided by NSGA-II.
    - SPEA-II and NSGA-III follow closely with cycle times of 53.62 s and 53.67 s, respectively.
3. **Robot Manoeuvrability**:
    - NSGA-III achieves the highest manoeuvrability index of 1.1786125, making it a viable choice for tasks requiring precise and agile robot movements.
    - While SPEA-II and NSGA-II also offer decent manoeuvrability scores with 1.1662671 and 1.1709163, respectively, they don't surpass NSGA-III in this regard.

In the overall assessment, the following observation can be made:

- If the primary goal is to minimise energy consumption or minimise the cycle time, then NSGA-II seems to be the most effective algorithm.
- To ensure the maximised manoeuvrability index, the NSGA-III is the best choice among these three algorithms.

In conclusion, the choice of algorithm should be based on the specific optimisation goals of the robotic application. While each algorithm has its strengths, the decision should align with the operational priorities to achieve the desired performance outcomes.

When focusing on minimised energy consumption, the NSGA-II is chosen, and the minimised energy consumption is 5613.59 wh. The energy usage reduces by 8.9% compared with the baseline of $6,164.98$ Wh. Information on the robot's pose under these conditions is depicted in Figure 8.33.

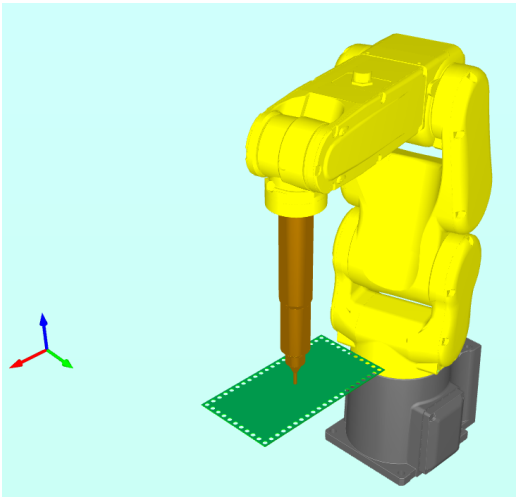


Figure 8.33: Pareto solution (x_c: 302.96, y_c: 76.83, z_c: 303.63) with minimised energy consumption

For minimised cycle time, a reduction of 6.0% from the baseline of 57.11 s is achieved, resulting in a cycle time of 53.15 s. The corresponding robot pose is illustrated in Figure 8.34.

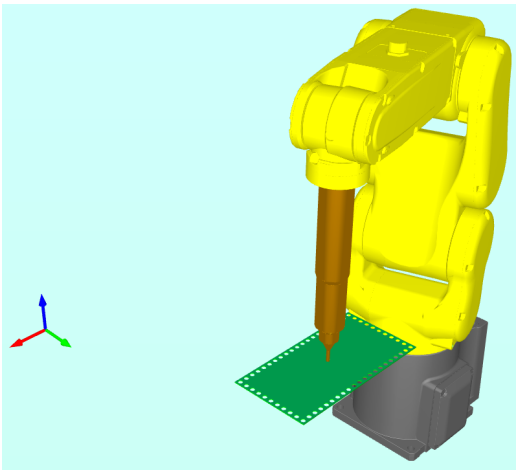


Figure 8.34: Pareto solution (x_c: 300.34, y_c: 87.62, z_c: 303.63) with minimised cycle time

Regarding the pursuit of a maximised robot manoeuvrability index, an increase of 140.8%

from the baseline of 0.4891235 is achieved, resulting in a maximised value of 1.1786125. The associated robot and workpiece positions under these conditions are illustrated in Figure 8.35.

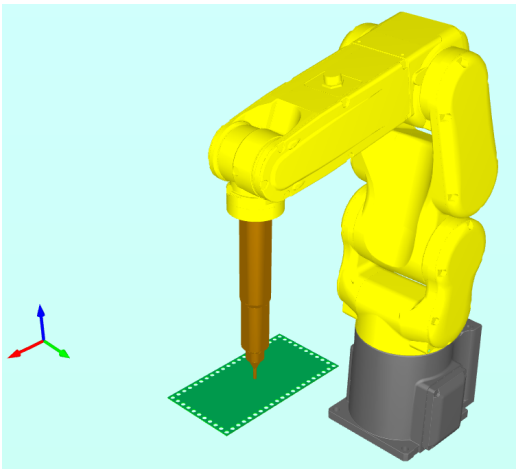


Figure 8.35: Pareto solution (x_c: 324.47, y_c: -0.12, z_c: 237.42) with maximised manoeuvrability index

8.2.4 Discussion of the Results of Use Case 2

The aim of Use Case 2 is to utilise the experience databank to optimise the layout of a single robot for a defined pose with the help of a digital twin. Asset selection is not heavily discussed in this use case. From the result, and according to the proposed methodology, the single robot's pose position is optimised based on multi-objectives. Table 8.12 describes the points that have been covered and validated in this use case.

Table 8.12: Validation according to the criteria in Use Case 2

			Validation
Contribution 1	Criterion 1.1	Vendor Neutrality	✓
	Criterion 1.2	Information Modelling	✓
	Criterion 1.3	Handling of Data from Diverse Sources	✓
	Criterion 1.4	Reasoning based on the Ontology	✓
Contribution 2	Criterion 2.1	Adapting to New Process Requirement	
	Criterion 2.2	Capability Assessment	
	Criterion 2.3	Modular Asset Selection	
	Criterion 2.4	Multi-Criteria Asset Selection	
	Criterion 2.5	Synergies with System	
Contribution 3	Criterion 3.1	Multi-Criteria Layout Optimisation	✓
	Criterion 3.2	Modular Layout Optimisation	✓
	Criterion 3.3	Interoperability	✓
	Criterion 3.4	Scalability	

Several key criteria are meticulously assessed in this use case to ensure comprehensive validation of the methodology. A concise summary of this validation is presented below:

- **Criterion 1.1 (Vendor Neutrality):** The utilisation of the experience databank for layout optimisation validates this criterion. The databank, sourced from diverse origins such as robot-related technical documents and equipment data, ensures an unbiased, vendor-neutral approach.
- **Criterion 1.2 (Information Modelling):** Validation is evident in the experience databank, pivotal for optimisation, effectively modelling diverse data types.
- **Criterion 1.3 (Handling of Data from Diverse Sources):** This criterion is enhanced by the experience databank's capability to integrate data from heterogeneous sources, illustrating the methodology's proficiency in managing diverse datasets.
- **Criterion 1.4 (Reasoning Based on the Ontology):** The ability of the methodology to update the experience databank dynamically and perform subsequent reasoning demonstrates the validity of this criterion.
- **Criterion 3.1 (Multi-Criteria Layout Optimisation):** The optimisation, influenced by multiple objectives such as cycle time, robot manoeuvrability, and energy consumption, ratifies this criterion.
- **Criterion 3.2 (Modular Layout Optimisation):** The adaptability in algorithm selection and the flexibility in changing decision variables affirm the modularity of the methodology, thus validating this criterion.
- **Criterion 3.3 (Interoperability):** The integration of the RoboDK API, ensuring seamless communication between the simulation software and the optimisation environment, combined with socket communication for real-time equipment data exchange, validates this criterion.

Significantly, **Criterion 3.4 (Scalability)** has been validated by optimising single assets in Use Case 2 and applying this optimisation to multiple assets in Use Case 1. Certain criteria, namely **Criterion 2.1 (Adapting to New Process Requirement)**, **Criterion 2.2 (Capability Assessment)**, **Criterion 2.3 (Modular Asset Selection)**, **Criterion 2.4 (Multi-Criteria Asset Selection)**, and **Criterion 2.5 (Synergies with System)** are not the focal points of this use case and consequently, are not validated here.

To conclude, this use case provides an exhaustive validation of the methodology across multiple criteria, reinforcing its versatility and applicability in varied scenarios.

8.3 Use Case 3 – Modular Multi-Objective Asset Selection with Experience Databank

As highlighted in Use Case 1 and Use Case 2, specific criteria remain unvalidated, specifically Criterion 2.3 (Modular Asset Selection), Criterion 2.4 (Multi-Criteria Asset Selection) and Criterion 2.5 (Synergies with System). To address this, Use Case 3 has been introduced. This case focuses on the aerospace manufacturing domain, emphasising the crucial role of the experience databank in optimising modular asset selection. The process involves identifying and evaluating potential assets, factoring in both explicit and implicit weight considerations across various algorithms and objectives. Key components of this use case can be seen in Figure 8.36.

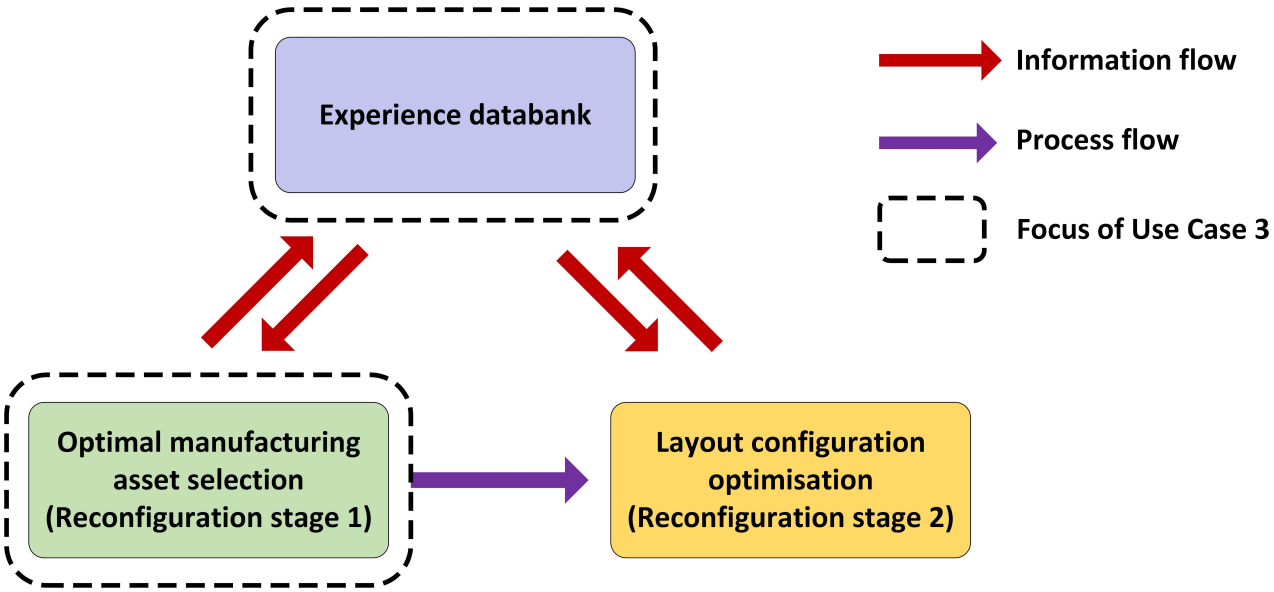


Figure 8.36: Focus of Use Case 3

8.3.1 Introduction

At the outset, the facility is preoccupied with fabricating a "Product I" component. Given evolving production requisites, a demand emerges to pivot towards "Product II". Such a tran-

sition mandates extensive recalibrations in the prevailing production process, encompassing alterations in operational protocols and the discerning integration of resources, notably robots and end effectors. Following the asset realignment, a subsequent phase of optimisation of the production cell is initiated to harmonise with the newly assimilated resources.

The extant production cell is segmented into two pivotal workstations:

- Workstation 1: Perform assembly tasks for the manufacturing equipment, which are later used for assembling and processing the product.
- Workstation 2: A versatile unit adept at product assembly and processing.

The emerging challenge arises with the change in production focus from "Product I" to "Product II", as this shift represents a transformation in process requirements. The subsequent goal is to validate the adaptability of the existing production cell in light of these new demands and, where necessary, initiate asset alterations or replacements. Following this assessment, the enhanced scenario necessitates careful fine-tuning to guarantee operational excellence, rooted in insights derived from a knowledge graph and an ontology model. The methodical execution of this use case proceeds as follows:

1. Developing the experience databank to adapt to changing process requirements.
2. Selecting assets using the information from the ontology model and the experience databank, identifying the best assets considering specification efficiency score and cost.

8.3.2 Generation of the Experience Databank

A dataset is curated to validate the proposed methodology, drawing on insights from the OMNIFACTORY demonstrator. Despite the inherent adaptability, discerning the optimal configuration for fabricating a novel or modified product remains an intricate undertaking.

The dataset harnessed for validation amalgamates technical literature, equipment specifications, and product blueprints sourced from OMNIFACTORY and its affiliate entities. This dataset catalogues 195 prospective assets, each with information that paves the way for capability congruence and the reconfiguration exercise. These assets span the gamut from pro-

duction line constituents to an expansive asset repository, encapsulating hardware, software, human capital, and reconfiguration algorithms, each playing a pivotal role in the production and recalibration processes. Analogous to Use Case 1, the imperative for data anonymisation is acknowledged in Use Case 3 as well. Tools such as Neo4j and Py2neo, as previously alluded to, facilitate the knowledge graph generation for this specific use case.

Given the extant production cell's commitment to "Product I", and in light of the introduction of "Product II", the task is to ascertain the feasibility of production line reconfiguration.

8.3.3 Modular Asset Selection Utilising Experience Databank

After the generation of the knowledge graph, the next step is to verify the approach proposed for asset selection. As per the knowledge graph generated, the preceding product, designated as "Product I" comprises two features, each necessitating different operations. Feature 1 requires a "Pick and Place" operation (Operation 1), a "Marking" operation (Operation 2), and a "Metrology" operation (Operation 3) at station 1. Feature 2 calls for a "Pick and Place" operation (Operation 4), a "Welding" operation (Operation 5), and a "Metrology" operation (Operation 6) at station 3. As per the proposed ontology model, the required capabilities for these six operations are all considered to be combined capabilities. Detailed information can be found in Table 8.13.

Table 8.13: Capability information of "Product I"

	Pick and place (Operation 1)	Marking (Operation 2)	Metrology (Operation 3)	Pick and place (Operation 4)	Welding (Operation 5)	Metrology (Operation 6)
Required Capability	Pick and place	Marking	Metrology	Pick and place	Welding	Metrology
Has Input Capability	Moving, ForceApplying, Releasing, Grasping	Moving, MarkingElement	Moving, MetrologyElement	Moving, ForceApplying, Releasing, Grasping	Moving, EjectFlame	Moving, MetrologyElement

Based on the information in Table 8.13, the ontology model and the current information on production, the detailed requirement can be described, as presented in Table 8.14.

Table 8.15 lists the current assets on the production line. At Station 1, the assets employed are KUKA KR 150 R3700 K ultra, marker 1, and V-stars metrology end effector 1. At Station 3, the related assets are the Fanuc R-2000iB/125L, Arc Welding End Effector, and V-Stars metrology end effector 2. The specifications of these assets are provided in Table 8.16.

Table 8.14: Process requirements for producing "Product I"

	Pick and place (Operation 1)	Marking (Operation 2)	Metrology (Operation 3)	Pick and place (Operation 4)	Welding (Operation 5)	Metrology (Operation 6)
Allowed Grasping-Force	100			100		
Required Grasping-Type	Finger Grasping			Finger Grasping		
Required Reachability	2,000	2,200	2,300	2,000	2,300	1,500
Required Payload	250	150	100	200	200	180
Required Repeatability	0.06	0.10	0.12	0.2	0.3	0.2
Required LateralOffset			10		20	30
Required MeasurementFrequency			50			60
Required Radius			10			20
Required Range			50			30
Required WeldingType					Arc Welding	
Required Resolution		2,000				
Required Accuracy		0.03				
Required FlameSize					80	
Required WeldingDepth					0.03	

Table 8.15: Current assets to execute the operations

	Pick and place (Operation 1)	Marking (Operation 2)	Metrology (Operation 3)	Pick and place (Operation 4)	Welding (Operation 5)	Metrology (Operation 6)
KUKA KR 150 R3700 K ultra	✓	✓	✓			
Finger gripper-1	✓			✓		
Marker-1		✓				
Metrology end effector 1			✓			
Fanuc R-2000iB/125L				✓	✓	✓
Finger gripper-2				✓		
Welding end effector-1					✓	
Metrology end effector 2						✓

8.3.3.1 Identifying the Candidate Assets

As previously mentioned, upon receiving a new customer request for the production of "Product II", the experience databank indicates that two new features must be developed. Feature 1 requires both "Pick and Place", "Marker", and "Metrology" operations, while Feature 2 demands

Table 8.16: Related specification information of the current assets

	KUKA KR 150 R3700 K ultra	Pick and place effector 1	Marker 1	Metrology effector 1	Fanuc R-2000iB/ 125L	Pick and place effector 2	Arc welding effector 1	Metrology effector 2
Allowed GraspingForce		100				200		
Payload	150				125			
Reachability	3,701				3,002			
Repeatability	0.06				0.3			
DrillingDepth			0.25					
Allowed HoleDiameter_-max			20					
Allowed HoleDiameter_-min			4					
LateralOffset				20				20
Measurement Frequency				60				80
Radius				15				20
Range				60				70
FlameSize							100	
WeldingDepth							0.05	

the incorporation of "Pick and Place", "Drilling", and "Metrology", operations as shown in Figure 8.37.

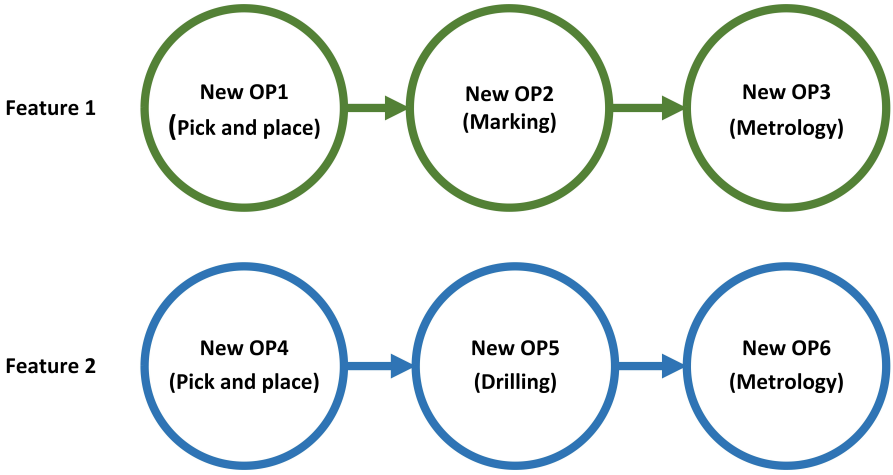


Figure 8.37: Feature requirement for the new product

The required capability and the description of the operations for the new product, according to the generated experience databank, are depicted in Table 8.17.

The detailed requirements for the new products are illustrated in Table 8.18. According to the recommendation from the experience databank, all the above operations described in the table require combined capabilities. The next section explains the asset selection process for all the required operations.

Table 8.17: Capability information of "Product II"

	Pick and place (New operation 1)	Marking (New operation 2)	Metrology (New operation 3)	Pick and place (New operation 4)	Drilling (New operation 5)	Metrology (New operation 6)
required capability	Pick and place	Marking	Metrology	Pick and place	Drilling	Metrology
has input capability	Moving, ForceApplying, Releasing, Grasping	Moving, MarkingElement	Moving, MetrologyElement	Moving, ForceApplying, Releasing, Grasping	Moving, SpinningTool, DrillBitFunction	Moving, MetrologyElement

8.3.3.1.1 PickAndPlace Operation (New Operation 1)

For Operation 1, PickAndPlace requires two assets: robots and grippers. The number of the candidate robot is 18, and the number of the candidate grippers is 26, as shown in Figure 8.38.

```
MATCH (:Task {Name:'Task 103'})-[:requiredProcess]→(d:Process
{feature:'F1', ProcessName:'PickAndPlace'})
WITH d
MATCH (x:Concept {name:d.ProcessName+'Capability'})-[r:SCO_RESTRICTION
{onPropertyName:"hasInputCapability"}]→(f:Concept)
CALL n10s.inference.nodesInCategory(f, {inCatRel:"InstanceCapability",
subCatRel:"SCO"}) YIELD node AS capability
WITH capability, d
MATCH (capability)-[:isTheCapabilityOf]→(dev:Assets)
WHERE dev.Payload > d.requiredPayload AND dev.Repeatability <
d.requiredRepeability AND dev.Reachability > d.requiredRepeability
RETURN count(DISTINCT dev) as assets_count
```

assets_count
18

Candidate robot for new operation 1

```
MATCH (:Task {Name:'Task 103'})-[:requiredProcess]→(d:Process
{feature:'F1', ProcessName:'PickAndPlace'})
WITH d
MATCH (x:Concept {name:d.ProcessName+'Capability'})-[r:SCO_RESTRICTION
{onPropertyName:"hasInputCapability"}]→(f:Concept)
CALL n10s.inference.nodesInCategory(f, {inCatRel:"InstanceCapability",
subCatRel:"SCO"}) YIELD node AS capability
WITH capability, d
MATCH (capability)-[:isTheCapabilityOf]→(dev:Assets)
WHERE dev.GraspingForce_max > d.requiredGraspingForce
RETURN count(DISTINCT dev) as assets_count
```

assets_count
26

Candidate gripper for new operation 1

Figure 8.38: Candidate assets information of the new Operation 1

8.3.3.1.2 Marking Operation (New Operation 2)

As described in the experience databank, executing the marking operation requires one combined capability called MarkingCapability, which consists of moving and marking element capabilities. The moving capability is executed by the asset with the type of asset. The marking element capability is executed by the asset with the type of marker and its subclasses, such as ink marker and stamper. The number of the candidate assets is presented in Figure 8.39.

8.3.3.1.3 Metrology Operation (New Operation 3)

As described in the experience databank, executing the metrology operation requires one combined capability called MetrologyCapability, which consists of moving and metrology element capability. The moving capability is executed by the asset with the type of robot. Metrology element capability is executed by the asset with the type of metrology device and its subclasses. The number of the candidate assets can be found in Figure 8.40.

Table 8.18: Process requirements for producing "Product II"

	Pick and place (New operation 1)	Marking (New operation 2)	Metrology (New operation 3)	Pick and place (New operation 4)	Drilling (New operation 5)	Metrology (New operation 6)
Allowed Grasping-Force	100			100		
Required Grasping-Type	Finger Grasping			Finger Grasping		
Required Reachability	2,000	2,200	2,300	2,000	2,300	1,500
Required Payload	180	30	20	200	200	180
Required Repeatability	0.2	0.2	0.15	0.2	0.3	0.2
Required LateralOffset			30			30
Required MeasurementFrequency			50			60
Required Radius			10			20
Required Range			50			30
Required Resolution		1,000				
Required Accuracy		0.03				
Required DrillingDepth					0.03	
Allowed HoleDiameter_max					0.4	
Allowed HoleDiameter_min					0.35	

8.3.3.1.4 PickAndPlace Operation (New Operation 4) As described in the experience databank, executing the PickAndPlace operation requires one combined capability called PickAndPlaceCapability, and this PickAndPlaceCapability consist of moving, force applying, releasing and grasping capabilities. Moving and force applying capabilities are executed by the asset with the type of robot. Grasping and releasing capabilities are executed by the asset with the type of Grippers. The number of the candidate assets can be found in Figure 8.41.

8.3.3.1.5 Drilling Operation (New Operation 5) As described in the experience databank, executing the drilling operation requires one combined capability called DrillingCapability, which consists of moving, spinning tools and DrillBitFunction capability. The moving capability is executed by the asset with the type of robot. The SpinningTool and DrillBitFunction capabilities are executed by the asset with the type of drilling tool. The number of candidate assets can be found in Figure 8.42.

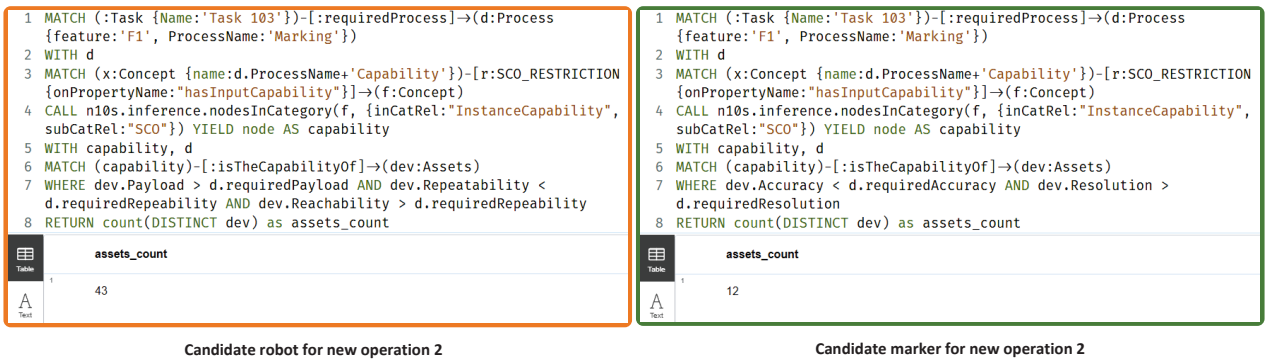


Figure 8.39: Candidate assets information of the new Operation 2

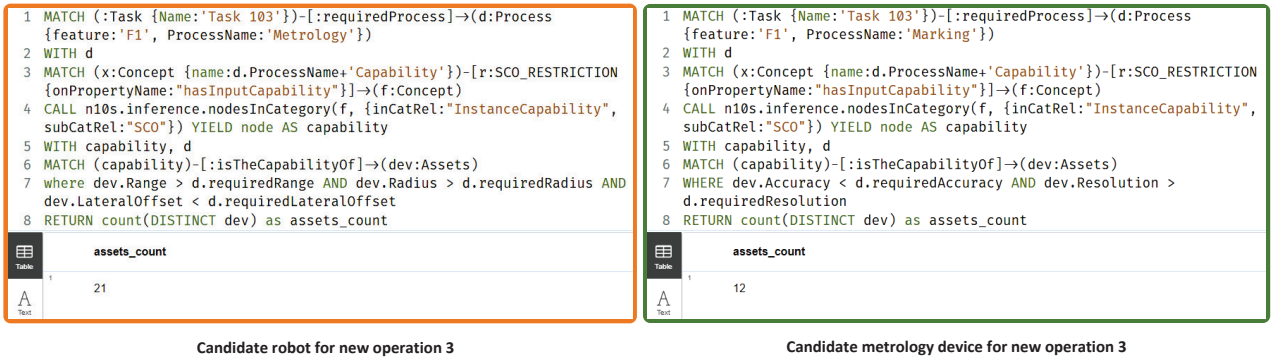


Figure 8.40: Candidate assets information of the new Operation 3

8.3.3.1.6 Metrology Operation (New Operation 6) As described in the experience databank, executing the metrology operation requires one combined capability called MetrologyCapability, which consists of moving and metrology element capability. The moving capability is executed by the asset with the type of robot. The metrology element capability is executed by the asset with the type of metrology device and its subclasses. The number of candidate assets can be found in Figure 8.43.

So, the total number of combinations of the candidate assets n for Product II can be calculated in Equation (8.15), yielding approximately $2.9 * 10^{15}$ possibilities. The proposed methodology aims to evaluate the combinations and find suitable combinations to adapt to the new process requirement based on the new product request.

$$n = 18 \times 26 \times 43 \times 12 \times 21 \times 12 \times 18 \times 26 \times 21 \times 21 \times 18 \times 19 \implies n \approx 2.9 \times 10^{15} \quad (8.15)$$

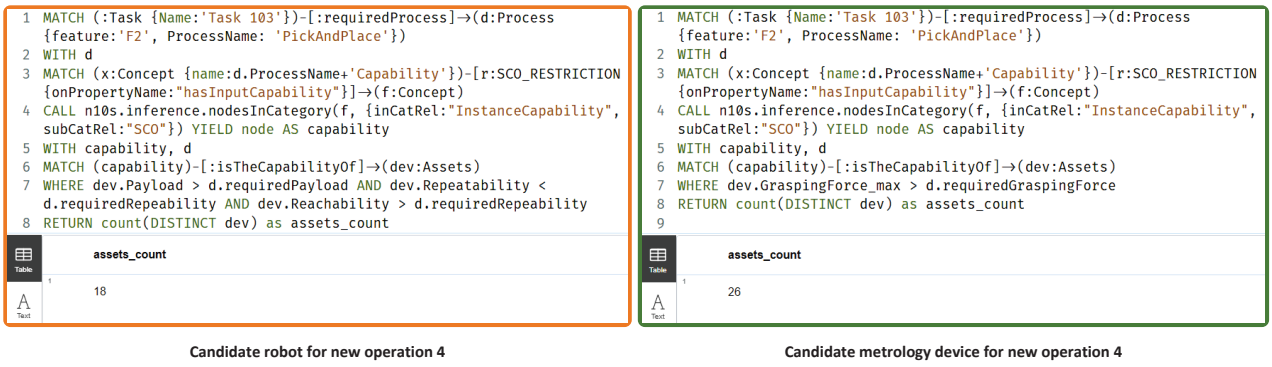


Figure 8.41: Candidate assets information of the new Operation 4

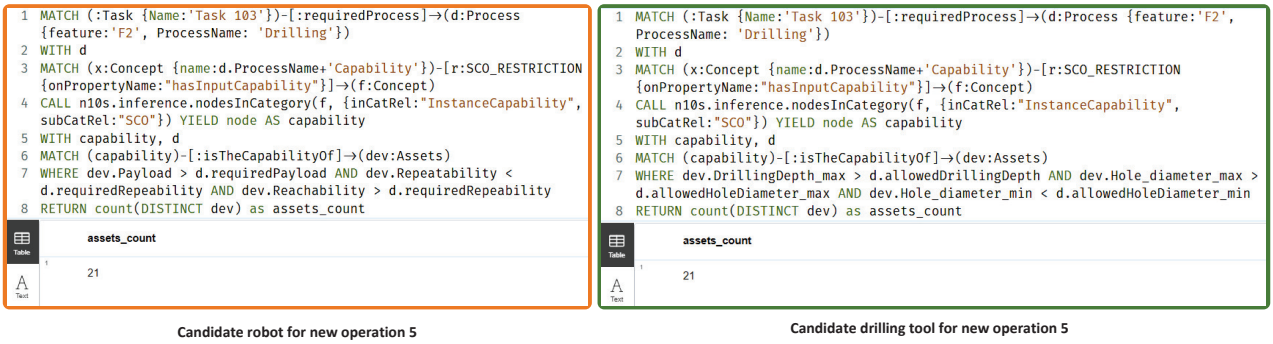


Figure 8.42: Candidate assets information of the new Operation 5

8.3.3.2 Candidate Asset Evaluation

This section details the method for selecting appropriate assets based on new customer requests. As outlined in the ontology model section, three major factors must be considered to address the resource selection issue: decision variables, constraints, and optimisation objectives.

8.3.3.2.1 Decision Variables In the proposed ontology model, decision variables are represented by resource information, the number of product types, product requirements, and job details. These elements are not merely standalone variables; they are further subdivided into subclasses within the semantic model to enhance their granularity and flexibility.

In real-world applications, these decision variables encompass the resource details of every potential asset and the index associated with each candidate group. An index list, denoted as $[a_1, a_2, \ldots, a_n]$, is constructed. This list represents an index for each candidate group, facilitating efficient asset identification and selection. Additionally, the index list can be employed to generate unique combinations of candidate assets, thereby ensuring that the requirements of new product requests are met.

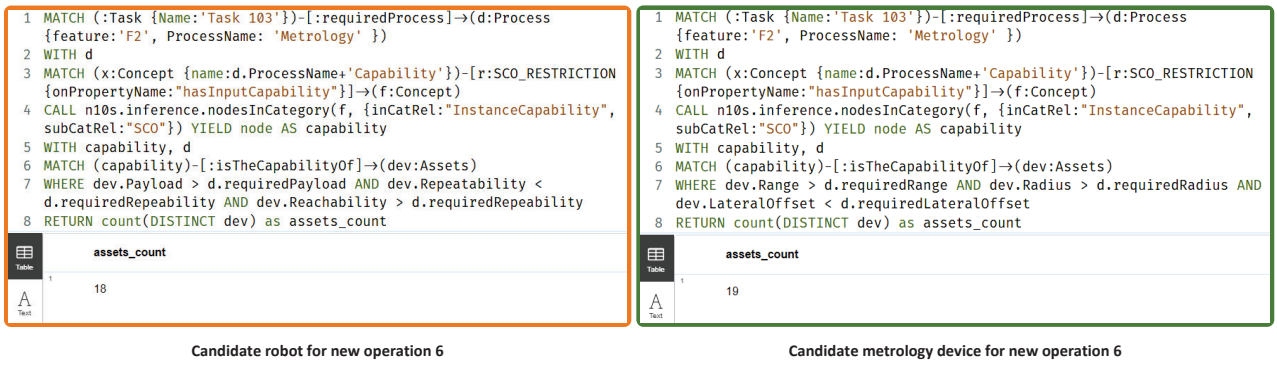


Figure 8.43: Candidate assets information of the new Operation 6

This methodology ensures a systematic approach to strategic decision-making in product development, grounded in a comprehensive analysis of resource availability, product types, requirements, and job specifics.

8.3.3.2.2 Constraints The primary task is to identify suitable assets for Workstations 1 and 2 in the existing production line. Each workstation is designed to operate with multiple robots, executing up to three operations. The optimal combination of assets should account for various cost factors, including installation, removal, and procurement. The following notation can be defined:

- SA: Set of all assets.
- SA_1: Set of assets for Workstation 1.
- SA_2: Set of assets for Workstation 2.
- O: Original asset list.
- C: Candidate asset list (newly suggested assets).
- $Cost_{Installation_a}$: Installation cost for asset a.
- $Cost_{Removal_a}$: Installation cost for asset a.
- $Cost_{Purchase_a}$: Installation cost for asset a.

Using the above notation, the constraints can be articulated as follows:

1. **Distinct Assets for Workstations**:

$$SA_1 \cap SA_2 = \emptyset \tag{8.16}$$

2. **Installation and Purchase Costs**: For every asset a that is on the candidate list but not on the original list, the costs for installation and, if not present in the facility, purchase are considered.

$$a \in C \setminus O \implies \begin{cases} \text{Consider } Cost_{Installation_a} \\ a \notin \text{facility} \implies \text{Consider } Cost_{Purchase_a} \end{cases} \tag{8.17}$$

3. **Removal Costs**: For every asset a that is on the original list but not on the candidate list, the removal cost is considered.

$$a \in O \setminus C \implies \text{Consider } Cost_{Removal_a} \tag{8.18}$$

4. **Relocation Costs**: If assets are present on both lists but have different workstation recommendations, both removal and installation costs should be considered.

5. **No Duplicate Costs**: Ensure that no asset's costs (installation, removal, and purchase) are counted multiple times.

8.3.3.2.3 Optimisation Objectives About selecting and evaluating the most appropriate assets for new customer requests in aerospace manufacturing, several factors should be considered when selecting evaluators for Product II. In this use case, the specification efficiency score and cost are chosen as the evaluators. This problem can be regarded as a combinatorial optimisation problem; for each operation, one of the candidate assets should be selected to meet the operation requirement. The problem is converted to find the best asset combination for all operational requirements while considering some optimisation objectives and constraints.

First, the specification efficiency score is considered, as it is crucial to prevent resource wastage due to overqualification, especially in the highly specialised field of aerospace manufacturing. Ensuring that the assets are tailored to the specific requirements of the new hinged product without unnecessary over-specification results in optimal resource utilisation.

Second, the cost is considered, encompassing the reconfiguration cost of the robotic assembly cell and the purchase cost if there are no available assets in the current databank. Aerospace manufacturing often necessitates high precision and quality; hence, balancing the costs of acquiring new assets with maintaining high production standards is vital. Ensuring the reconfiguration cost does not exceed the reconfiguration budget.

The optimisation objectives are depicted in Table 8.19.

Table 8.19: Optimisation objectives and targets

Optimisation Objectives	**Target**
Specification Efficiency Score	Maximise
Reconfiguration Cost	Minimise

Considering these evaluators, the production line will be better equipped to select the most suitable assets to meet the new customer requirement, Product II.

8.3.3.2.3.1 Maximising the Specification Efficiency Score To calculate the specification efficiency score, this score is defined for all four new operations and their related assets (robot + end effector). The specification evaluators for each operation are listed in Table 8.20.

Table 8.20: Specification evaluators for each operation

New Operations	**Robots**	**End effector**
New Operation 1	Payload, Reachability, Repeatability	GraspingForce
New Operation 2	Payload, Reachability, Repeatability	Resolution, Accuracy
New Operation 3	Payload, Reachability, Repeatability	LateralOffset, MeasurementFrequency, Radius, Range
New Operation 4	Payload, Reachability, Repeatability	GraspingForce
New Operation 5	Payload, Reachability, Repeatability	DrillingDepth, HoleDiameter_max, HoleDiameter_min
New Operation 6	Payload, Reachability, Repeatability	LateralOffset, MeasurementFrequency, Radius, Range

The specification efficiency score is the index related to different evaluators. For example, for the robot in NewOperation1, the sub-evaluators are payload, reachability, and repeatability. For the end effector in NewOperation1, the evaluator is GraspingForce. Obtaining information about the relationship between different evaluators of the same asset in the same operation

is essential. If the weights of the involved sub-evaluators are known, then the specification efficiency score is a single objective to optimise. In the current use case, the weights of each evaluator are assumed to be the same. The optimisation problem thus aligns with the description provided in Table 8.19, referencing the methodology delineated in Section 5.4.3.3.

8.3.3.2.3.2 Minimising the Cost To calculate the cost, different costs should be considered. The total cost of reconfiguration to meet the new customer request can be calculated using Equation (5.18). The reconfiguration cost can be calculated as shown in Equation (8.19):

$$Cost_{Reconfiguration} = \sum_{i=1}^{a} Cost_{Installation_i} + \sum_{j=1}^{b} Cost_{Removal_j} + \sum_{k=1}^{c} Cost_{Purchase_k}, \tag{8.19}$$

where a is the number of assets that must be installed and, b is the number of assets which need to be removed, c is the number of assets which need to be purchased.

The process for determining the installation, purchase, and removal costs is depicted in Figure 8.44.

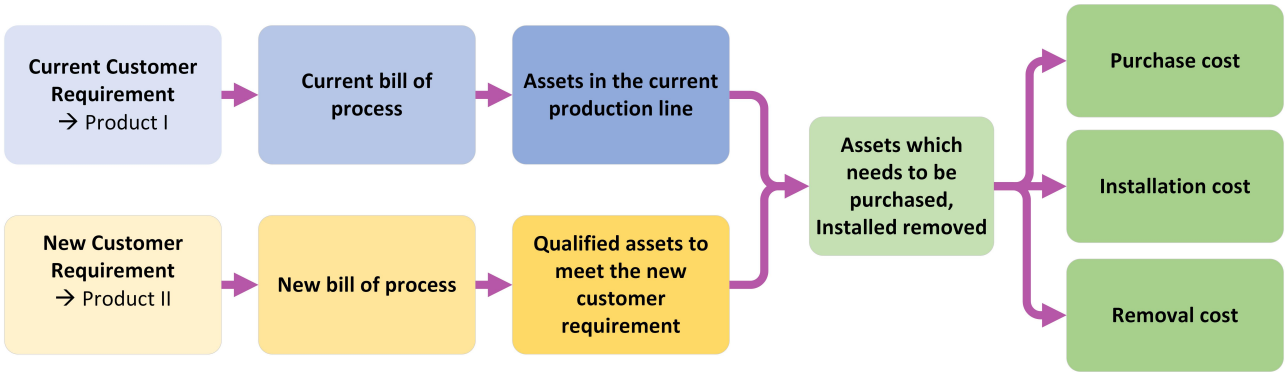


Figure 8.44: Process for acquiring cost-related information

8.3.3.2.4 Evaluation To solve the above multi-objective optimisation problem, the experience databank recommends the use of NSGA-II and SPEA-II. These multi-objective optimisation algorithms are employed to carry out the optimisation and asset selection processes. The hyperparameters of NSGA-II and SPEA-II are depicted in Table 8.21.

The optimisation curve of the NSGA-II algorithm for the two objectives is presented in Figure 8.45. The data indicates that for the single-objective cost, the minimal value identified is

Table 8.21: Comparison of hyperparameters in NSGA-II and SPEA-II

Hyperparameter	NSGA-II	SPEA-II
Population size	500	500
Sampling method	IntegerRandomSampling	IntegerRandomSampling
Crossover method	SBX with eta=20 and prob=0.5	SBX with eta=20 and prob=0.5
Mutation method	Polynomial Mutation with eta=20	Polynomial Mutation with eta=20
Generation	500	500

£21,090. The maximum specification efficiency score recorded is 0.831.

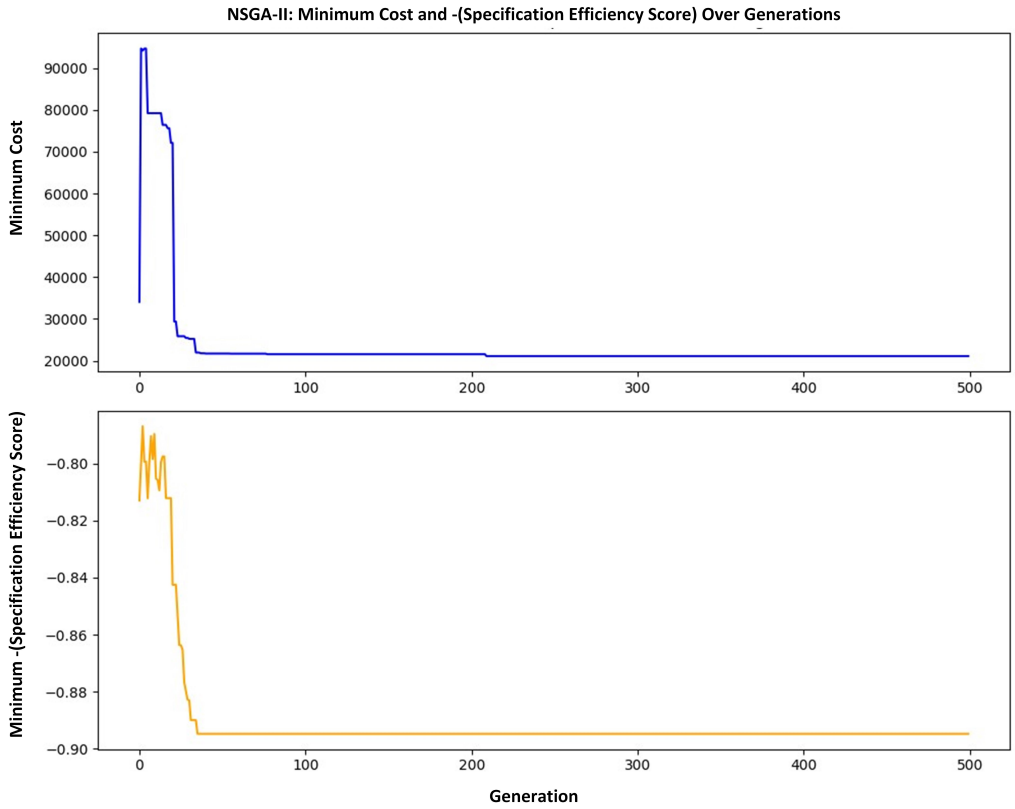


Figure 8.45: Optimisation result of NSGA-II

The Pareto front it can find is depicted in Figure 8.46, and some Pareto-optimal solutions can also be found. This study aims to find the Pareto front solution with the lowest cost. The red point denotes the Pareto solution with the lowest cost needed. At that point, the cost is £21,090, and the specification efficiency score is 0.8306. The corresponding information for this solution is listed in Table 8.22. The specification efficiency score information is depicted in Table 8.23.

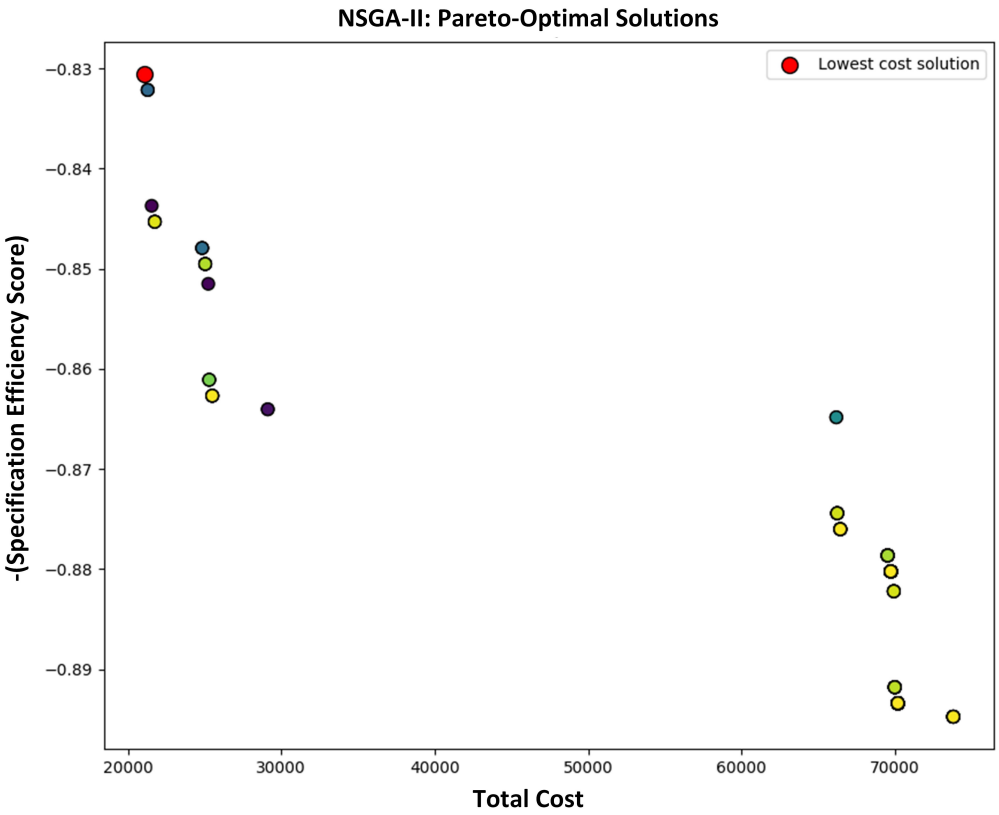


Figure 8.46: Pareto solutions of NSGA-II

Table 8.22: Reconfiguration cost details of NSGA-II

Candidate Assets	Cost type	Cost	Workstation
ABB IRB 6700-245/3.00	Installation	3,300	Workstation 1
FingerGripper-11	Installation	320	Workstation 1
Marker-9	Purchase	3,700	Workstation 1
Marker-9	Installation	460	Workstation 1
Metrology end effector 3	Purchase	4,100	Workstation 1
Metrology end effector 3	Installation	550	Workstation 1
Fanuc M-900ia/350	Installation	3,500	Workstation 2
Drilling Tool-1	Installation	300	Workstation 2
KUKA KR 150 R3700 K ultra	Removal	2,700	Workstation 1
Fanuc R-2000iB/125L	Removal	2,010	Workstation 2
WeldingEndEffector-1	Removal	150	Workstation 2

The optimisation curve for the two objectives is generated using SPEA-II, as depicted in Figure 8.47. Analysis reveals that, for the single-objective cost, the lowest identified cost is £20,920. Additionally, the highest specification efficiency score recorded is 0.865.

Table 8.23: Specification efficiency score for the solution found by NSGA-II

Operations	Updated Assets	Specification Efficiency Score	Workstation
NewOperation 1	ABB IRB 6700-245/3.00 FingerGripper-11	0.0735 0.0573	Workstation 1 Workstation 1
NewOperation 2	ABB IRB 6700-245/3.00 Marker-9	0.0484 0.0973	Workstation 1 Workstation 1
NewOperation 3	ABB IRB 6700-245/3.00 Metrology end effector 3	0.0503 0.0550	Workstation 1 Workstation 1
NewOperation 4	Fanuc M-900iA/350 FingerGripper-11	0.0913 0.0572	Workstation 2 Workstation 2
NewOperation 5	Fanuc M-900iA/350 Drilling Tool-1	0.0793 0.0668	Workstation 2 Workstation 2
NewOperation 6	Fanuc M-900iA/350 Metrology end effector 3	0.0913 0.0628	Workstation 2 Workstation 2

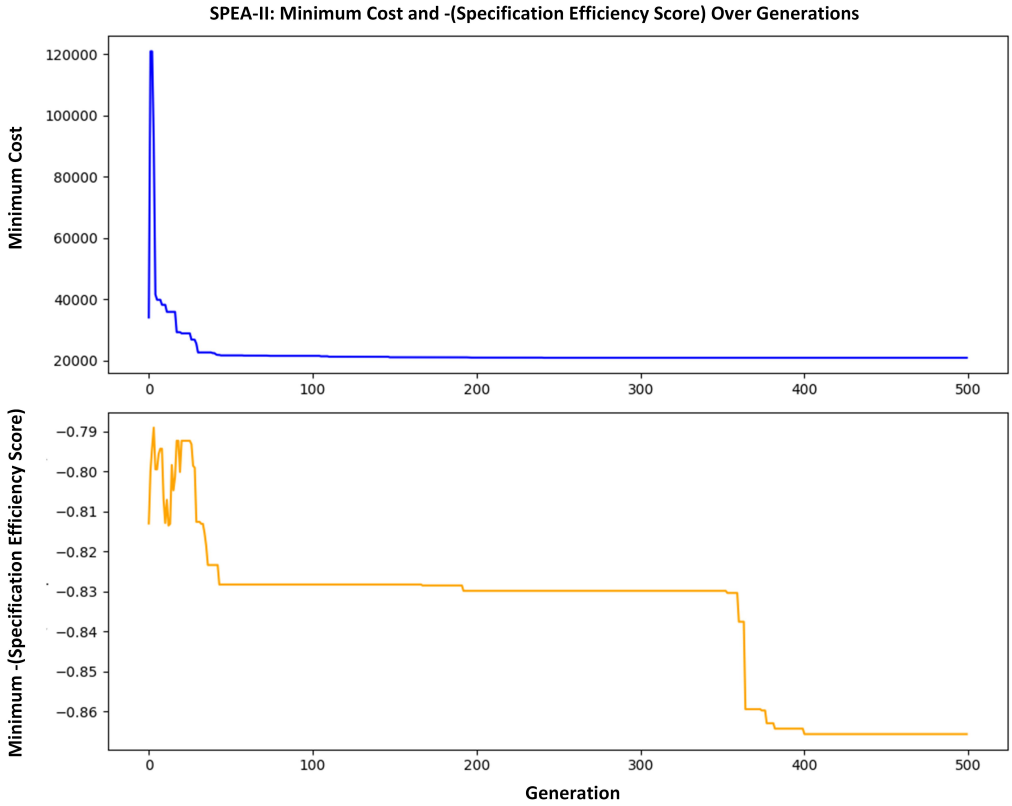


Figure 8.47: Optimisation result of SPEA-II

The Pareto front identified is illustrated in Figure 8.48. Several Pareto-optimal solutions are also discerned, and the objective is to identify the solution with the lowest cost. The red point represents the Pareto solution with the minimal cost of interest. At that point, the solution's cost is £20,920, and the specification efficiency score is 0.8146. The corresponding information for this Pareto solution is listed in Table 8.24, and information on the specification efficiency

score is depicted in Table 8.25.

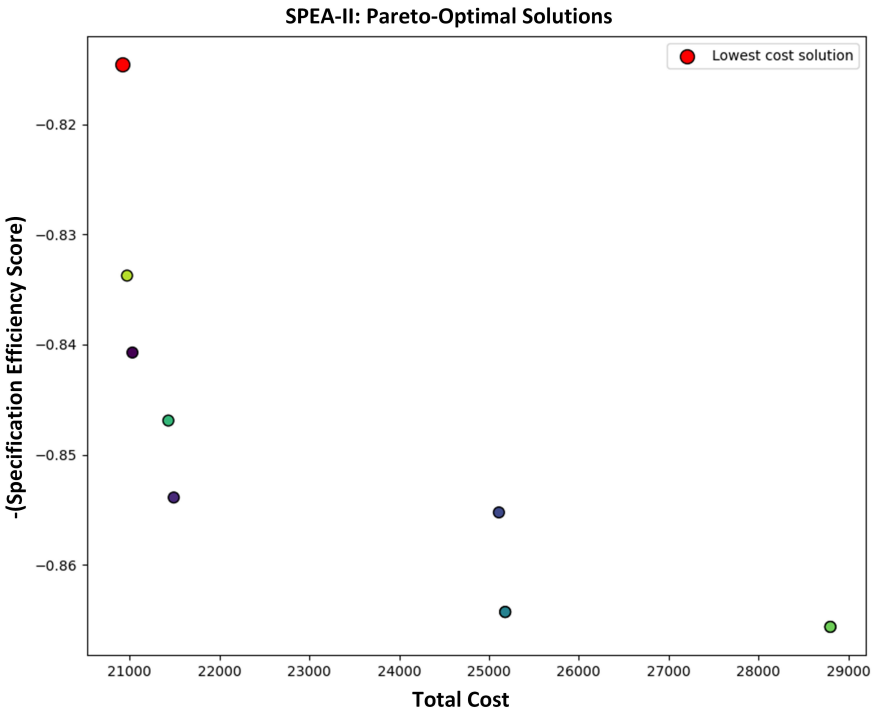


Figure 8.48: Pareto solutions of SPEA-II

Table 8.24: Reconfiguration cost details of SPEA-II

Candidate Assets	Cost type	Cost	Workstation
ABB IRB 6700-245/3.00	Installation	3,300	Workstation 1
FingerGripper-1	Installation	150	Workstation 1
Marker-9	Purchase	3,700	Workstation 1
Marker-9	Installation	460	Workstation 1
Metrology end effector 3	Purchase	4,100	Workstation 1
Metrology end effector 3	Installation	550	Workstation 1
Fanuc M-900ia/350	Installation	3,500	Workstation 2
Drilling Tool-1	Installation	300	Workstation 2
KUKA KR 150 R3700 K ultra	Removal	2,700	Workstation 1
Fanuc R-2000iB/125L	Removal	2,010	Workstation 2
WeldingEndEffector-1	Removal	150	Workstation 2

In light of the evaluations conducted using the two multi-objective optimisation algorithms, NSGA-II and SPEA-II, the findings reveal interesting implications for selecting and optimising robotic assets in response to new customer requests.

Table 8.25: Specification efficiency score for the solution found by SPEA-II

Operations	Updated Assets	Specification efficiency score	Workstation
NewOperation 1	ABB IRB 6700-245/3.00 FingerGripper-1	0.0735 0.0493	Workstation 1 Workstation 1
NewOperation 2	ABB IRB 6700-245/3.00 Marker-9	0.0484 0.0973	Workstation 1 Workstation 1
NewOperation 3	ABB IRB 6700-245/3.00 Metrology end effector 3	0.0503 0.0550	Workstation 1 Workstation 1
NewOperation 4	Fanuc M-900iA/350 FingerGripper-1	0.0913 0.0493	Workstation 2 Workstation 2
NewOperation 5	Fanuc M-900iA/350 Drilling Tool-1	0.0793 0.0668	Workstation 2 Workstation 2
NewOperation 6	Fanuc M-900iA/350 Metrology end effector 3	0.0913 0.0628	Workstation 2 Workstation 2

SPEA-II identified a Pareto solution with a lower cost than NSGA-II did (£20,920 vs. £21,090) and a reasonably competitive specification efficiency score (0.865 vs. 0.879). Hence, SPEA-II is selected to optimise robotic asset selection based on the comparative evaluation because it can find the most optimised cost. However, if the customer requires the most optimised asset with a higher specification efficiency score, then NSGA-II would be selected.

8.3.4 Discussion of the Results of Use Case 3

Use Case 3 utilises the experience databank to perform asset selection and evaluation in more complicated scenarios. With the help of the proposed experience databank, the asset selection can be executed clearly based on the new process requirement. The experience databank enhances the asset evaluation process, and it is modular. An effective solution asset evaluation method is executed, and the effective solution is found. The criteria in Table 8.26, with check marks for the objectives, are validated.

In Use Case 3, the emphasis is on validating various criteria to fulfil Objectives 1 and 2. The initial generated experience databank plays a fundamental role in enabling asset selection and evaluation. The criteria validated and the justifications for each are as follows:

- **Criterion 1.1 (Vendor Neutrality):** The use of the OCCR ontology model to build the experience databank validates this criterion.
- **Criterion 1.2 (Information Modelling):** This criterion is validated through the mod-

Table 8.26: Validation according to the criteria in Use Case 3

			Validation
Contribution 1	Criterion 1.1	Vendor Neutrality	✓
	Criterion 1.2	Information Modelling	✓
	Criterion 1.3	Handling of Data from Diverse Sources	✓
	Criterion 1.4	Reasoning Based on the Ontology	✓
Contribution 2	Criterion 2.1	Adapting to New Process Requirement	✓
	Criterion 2.2	Capability Assessment	✓
	Criterion 2.3	Modular Asset Selection	✓
	Criterion 2.4	Multi-Criteria Asset Selection	✓
	Criterion 2.5	Synergies with System	✓
Contribution 3	Criterion 3.1	Multi-Criteria Layout Optimisation	
	Criterion 3.2	Modular Layout Optimisation	
	Criterion 3.3	Interoperability	
	Criterion 3.4	Scalability	

elling of information regarding robotic assembly reconfiguration.

- **Criterion 1.3 (Handling of Data from Diverse Sources):** Given that data from various sources, such as technical documents, process requirements, and asset information, are incorporated, this criterion is validated.

- **Criterion 1.4 (Reasoning based on the Ontology):** Ontology reasoning is employed to identify potential assets that meet the process requirements, thus validating this criterion.

- **Criterion 2.1 (Adapting to New Process Requirement):** In this use case, the transition from addressing the process requirements of "Product I" to those of "Product II" is handled effectively, and this criterion is hence validated.

- **Criterion 2.2 (Capability Assessment):** The process of analysing capability and performing capability matching in this use case results in the validation of this criterion.

- **Criterion 2.3 (Modular Asset Selection):** The deployment of different algorithms (NSGA-II and SPEA-II) for the same task validates this criterion.

- **Criterion 2.4 (Multi-Criteria Asset Selection):** The use of multiple criteria, such as specification efficiency score and reconfiguration cost in this use case, establishes the validity of this criterion.

- **Criterion 2.5 (Synergies with Systems):** In asset selection, synergies with other assets prove critical. In this use case, the specification efficiency scores of various assets are evaluated collectively for asset selection. Furthermore, the cumulative cost of all assets

to fulfil varying process requirements is considered, thereby validating the synergies with the system.

8.4 Chapter Summary

This chapter builds upon the validation work initiated in Chapter 7, further contributing to the achievement of Objective 4. It presents a concise validation of the proposed methodology via three use cases, each addressing different scenarios. The integration of this chapter's findings with the software development insights from Chapter 7 completes the validation framework. The specific validation criteria for each use case are itemised in Table 8.27.

Table 8.27: Summary of validated criteria in different use cases

Criterion	Description	Use Case 1	Use Case 2	Use Case 3
Criterion 1.1	Vendor Neutrality	✓	✓	✓
Criterion 1.2	Information Modelling	✓	✓	✓
Criterion 1.3	Handling of Data from Diverse Sources	✓	✓	✓
Criterion 1.4	Reasoning Based on Ontology	✓	✓	✓
Criterion 2.1	Adapting to New Process Requirement	✓		✓
Criterion 2.2	Capability Assessment	✓		✓
Criterion 2.3	Modular Asset Selection			✓
Criterion 2.4	Multi-Criteria Asset Selection			✓
Criterion 2.5	Synergies with System			✓
Criterion 3.1	Multi-Criteria Layout Optimisation		✓	
Criterion 3.2	Modular Layout Optimisation		✓	
Criterion 3.3	Interoperability	✓	✓	
Criterion 3.4	Scalability	⋆	⋆	

Note: ⋆ indicates combined validation from Use Case 1 and Use Case 2.

Use Case 1 is tailored to validate the procedure for constructing and updating the experience databank, the asset selection methodology, and the layout optimisation framework. However, it primarily covers the system level, leaving the machine level unexplored. Its focus on single-objective layout optimisation also highlights potential questions about the scalability of the methodology.

To address these gaps, Use Case 2 addresses the machine level and expands the lens on layout optimisation by integrating multi-criteria considerations. Through the use of different types of simulation software in this scenario, the adaptability of the methodology is examined, adding rigour to the validation.

Use Case 3 redirects the focus, highlighting a unique aspect of the asset selection methodology. It accentuates the need for a holistic asset selection approach in expansive scenarios and

integrates a multi-objective framework. Both the modularity of the asset selection framework and its multi-objective aspects receive validation.

Collectively, these use cases provide a comprehensive validation approach. Various levels, software tools, and objectives meticulously test the adaptability, versatility, and depth of the methodology. This multi-pronged strategy ensures the scalability of the methodology across various situations.

Chapter 9

Conclusions and Future Work

The increasing complexity of modern manufacturing systems brings new challenges and opportunities. Quick and efficient adaptability is crucial in an era with fast market changes, unpredictable supply chains, and varying demand. This highlights the need to adjust and improve production processes skillfully.

Robots play a crucial and expanding role in the manufacturing process, making them essential for reconfiguration tasks. Robots are no longer merely tools but vital components of the production system that require careful management.

In this context, there is a pressing need to reconfigure robots within a robotic assembly system effectively. As their role in manufacturing tasks expands, efficiently harnessing their capabilities and adapting them to evolving requirements become essential. This central theme shaped this PhD research, which explores integrating robots within the broader context of refining and adjusting manufacturing processes.

9.1 Conclusions

This thesis contributes to the field of robotic assembly systems, focusing on the reconfiguration of these systems to keep pace with the changing demands of advanced manufacturing. The research addresses essential challenges in reconfiguring these systems, which are vital for sustaining operational efficiency and competitiveness in a dynamic industrial setting. The key

challenges explored in this study include:

1. Robotic assembly cells face the challenging task of absorbing and processing diverse data streams from multiple systems and technologies. This is crucial in order to consolidate the data into a coherent format that enables real-time decision-making.
2. The second significant challenge is the requirement for systems to continuously adapt to changing process and product requirements, reflecting the dynamic nature of market demands. The capacity to promptly and effectively adjust production processes in alignment with these changing needs is essential for sustaining a competitive edge.
3. The third challenge entails enhancing operational efficiency through the optimisation of layouts in robotic assembly cells, particularly after these systems are adapted to new process requirements. This involves identifying the optimal times to execute layout improvements and ensuring the framework remains scalable and adaptable to evolving demands.

The research questions for this PhD thesis have emerged from these three pivotal challenges. They are as follows:

1. RQ1

   How can data from various sources be efficiently processed by integrating diverse systems and technologies into robotic assembly cells?
2. RQ2

   How can robotic assembly cells adeptly adjust to ever-shifting process requirements, reflecting the changing consumer market?
3. RQ3

   How can a reconfigurable robotic assembly system efficiently optimise its operations, especially in layout, after adapting to current process requirements?

In alignment with these research questions, four objectives have been established:

- Objective 1: To develop an integrative approach for effectively managing heterogeneous manufacturing data within robotic assembly cells, which addresses RQ1 and supports informed decision-making.

- Objective 2: To design a reconfigurable robotic assembly cell system that exhibits agility and responsiveness to changing market demands and product requirements, thereby answering RQ2.
- Objective 3: To formulate a post-adaptation optimisation process for robotic assembly cells, focusing on layout optimisation using artificial intelligence, knowledge graphs, and simulation methodologies, in response to RQ3.
- Objective 4: To validate the strategies of the preceding three objectives through software development and testing within use cases.

To answer RQ1 and fulfil Objective 1, a methodology for aggregating and updating diverse data through an experience databank and ontology model, enhancing decision-making within robotic assembly cells, has been established. (Contribution 1)

The approach combines an ontology model and a knowledge graph method. The ontology model offers a vendor-neutral representation, making sense of capabilities and reconfiguration aspects, while the knowledge graph improves process efficiency. Merging both top-down and bottom-up approaches ensures accurate data representation and swift processing. A critical feature that signifies the system's adaptability is its ability to accommodate real-time queries. This synergy between the ontology model and the knowledge graph provides a robust solution for interpreting robotic assembly data, and the experience databank is built based on this. Thus, the RQ1 is addressed, and Objective 1 is fulfilled.

To answer RQ2 and fulfil Objective 2, a methodology has been formulated to facilitate modular asset selection, integrating knowledge graphs and multi-criteria decision-making algorithms, enabling rapid adaptation to changing manufacturing requirements. (Contribution 2)

It emphasises the importance of factors such as cost, energy consumption, asset longevity, adaptability over time, and resource utilisation. Acknowledging the evolving nature of these factors, the chapter highlights the necessity of a dynamic approach to asset selection. Advanced strategies such as multi-objective optimisation are introduced to navigate the multi-faceted challenges of this problem. These techniques adeptly manage both discrete and continuous decision variables, enhancing the precision and effectiveness of the selection process. In essence, the chapter offers a blueprint for manufacturers, equipping them with methodologies to refine asset selection and achieve greater agility in the ever-shifting manufacturing landscape.

To answer RQ3 and fulfil Objective 3, a methodology has been developed to enhance operational efficiency through layout optimisation in robotic assembly cells, incorporating simulation tools and multi-objective decision-making, facilitating continuous process improvements post-adaptation to changing requirements. (Contribution 3)

It focuses on developing and applying a modular optimisation framework for reconfigurable manufacturing systems, especially robotic assembly cells. This framework is built around three main components: the simulation environment, the optimisation environment, and the optimisation algorithms. A central aspect is the role of layout optimisation in robotic assembly cells to ensure the layout functions efficiently and within the constraints of the manufacturing environment. The inclusion of the experience databank aids in informed decision-making based on past data. This research showcases the benefits of using such knowledge-based systems in modern manufacturing design. This contribution presents a systematic approach to creating flexible, efficient, and durable manufacturing solutions that adapt to changing market conditions and customer needs.

To fulfil Objective 4, the realisation of the software suite comprising experienced databank, asset selection, and layout optimisation has been achieved, and three use cases have been proposed. This validation demonstrates that the impact of Objective 4 is not solely dependent on the software's functionality but also on its proven applicability within real-world industrial scenarios. (Contribution 4)

The software interface for the experience databank, designed as a central repository, leverages ontology models and knowledge graph methodologies. This design supports Contribution 1 by enhancing decision-making within robotic assembly cells. For Contribution 2, a specialised software tool has been developed for asset selection. This tool, utilising decision matrices and algorithms based on knowledge graphs and multi-criteria decision-making techniques, enables a flexible asset selection process. It considers varying parameters such as specification efficiency score, energy consumption, and robot manoeuvrability. To support Contribution 3, a sophisticated software framework for layout configuration optimisation has been implemented. It integrates simulation tools like Tecnomatix Process Simulate and RoboDK, alongside multi-objective optimisation algorithms. This framework, drawing insights from the experience databank, continually improves the layout of robotic assembly cells.

The effectiveness and relevance of these software tools have been rigorously evaluated through

three distinct use cases. Use Case 1 demonstrates the system-level application of the experience databank, asset selection methodology, and layout optimisation framework. Use Case 2 explores machine-level intricacies, employing detailed multi-criteria evaluations and highlighting the flexibility of the layout optimisation approach with various simulation tools. Use Case 3 emphasises the modular and comprehensive nature of the asset selection methodology through a multi-objective framework. Use Case 1, Use Case 2, and Use Case 3 collectively serve to validate the robustness, scalability, and industrial relevance of the contributions, demonstrating their suitability for practical scenarios. This comprehensive validation confirms that the methodologies developed are academically sound and ready to enhance the adaptability, efficiency, and resilience of robotic assembly systems, addressing the manufacturing sector's evolving challenges.

9.2 Applications in the Manufacturing Industry

The significant developments in the manufacturing sector highlight the importance and timeliness of this research's methodologies and findings. As industries adapt to modern challenges, the insights from this study serve as essential guidelines.

9.2.1 Automotive Industry

In contemporary manufacturing, particularly within the automotive sector, the establishment of an experience databank stands as a precursor to informed decision-making. This thesis asserts the databank's role as the cornerstone in the aggregation of knowledge — a repository that meticulously records outcomes and processes, setting the stage for successive optimisations.

Upon the foundation laid by the experience databank, asset selection is approached with an empirical richness. The detailed records of past performance and operational data guide the selection of robotic assets, ensuring that the choice is not merely theoretical but grounded in historical efficacy. Especially within the realm of electric vehicle (EV) production, where novel challenges, such as the assembly of delicate battery systems and integration of complex electrical components, are encountered, this informed approach to asset selection is invaluable. The vendor-neutral ontology model introduced by this thesis further refines this process, facilitating

the integration of diverse data sources and the discernment of the most apt robotic systems for the task at hand.

Once the assets are selected, the research progresses to address the spatial dynamics of manufacturing through layout optimisation. In the transition to EV production, where traditional manufacturing layouts must be rethought to accommodate new processes, this phase gains amplified significance. The modular layout configuration framework proposed herein enables manufacturers to rapidly reconfigure assembly lines in response to the ebb and flow of market demand and technological evolution. Such an adaptive approach to layout optimisation is not just about spatial economy but is fundamental to maintaining productivity and flexibility in an industry undergoing a significant paradigm shift towards electrification.

Therefore, it becomes apparent that the systematic approach advocated by this research, starting from the creation of an experience databank through to the meticulous selection of assets and culminating in the strategic optimisation of layout, provides a comprehensive blueprint for the automotive industry. It equips manufacturers with the necessary tools to navigate the intricacies of EV manufacturing, heralding a future where responsiveness and efficiency are harmoniously balanced.

9.2.2 Aerospace Manufacturing

In aerospace manufacturing, where precision and complexity define the sector, the establishment of a vendor-neutral ontology model as the initial phase of integration is pivotal. This model centralises the wealth of data from avionics, propulsion, and structural domains, thereby enabling a nuanced approach to asset selection. Given the sector's reliance on high-precision robotics to perform tasks ranging from turbine blade fabrication to fuselage assembly, the selection of these assets is not merely a matter of choice but of strategic necessity.

Following the initial step of data integration and asset selection, attention turns to the optimisation of manufacturing layouts. In an industry characterised by large-scale assembly lines and complex component integration, the configuration of space is of the essence. The challenge here differs from the automotive sector as it requires accommodating the assembly of massive structures and the precise coordination of numerous tasks that are often unique to each aircraft model. The layout optimisation must not only allow for efficient space utilisation but

also be flexible enough to accommodate customisations and the incorporation of technological advancements as they emerge.

This research underlines the importance of a strategic sequence in aerospace manufacturing, starting with the consolidation of data that informs the selection of robotic assets, and progressing to the systematic arrangement of manufacturing workflows. The methodologies espoused provide a framework that ensures aerospace manufacturers can maintain precision and adaptability in a domain where the costs of error are high and the demands for innovation are relentless. Thus, while sharing a common thread with automotive industry applications in terms of process structure, the application in aerospace manufacturing distinguishes itself through its focus on precision and customisation at an expansive scale.

9.3 Future Work

Future investigations will extend the optimisation objectives to enrich criteria for selecting optimal assets and improving layout configuration. The framework will aim to incorporate manufacturing standards, with a particular focus on adopting the Reference Architectural Model Industrie (RAMI) for delineating manufacturing knowledge [102]. The intention is to compile and disseminate a substantial dataset relevant to manufacturing reconfiguration scenarios. This dataset will support extended research into knowledge graphs, link prediction [122], recommendation systems [31], and reinforcement learning techniques in knowledge graph inference [123]. Additionally, future work will explore the synergy between control reconfiguration and the three identified categories of reconfiguration in this thesis, with the goal of integrating these dimensions into a holistic reconfiguration model, thus enriching the framework's depth and utility.

Acknowledgement

This research is supported by the DiManD Innovative Training Network (ITN) project funded by the European Union through the Marie Skłodowska-Curie Innovative Training Networks (H2020-MSCA-ITN-2018) under grant agreement number no. 814078.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 814078

References

[1] Benny Tjahjono, C Esplugues, Enrique Ares, and G Pelaez. What Does Industry 4.0 Mean to Supply Chain? *Procedia manufacturing*, 13:1175–1182, 2017.

[2] Judit Nagy, Judit Oláh, Edina Erdei, Domician Máté, and József Popp. The Role and Impact of Industry 4.0 and the Internet of Things on the Business Strategy of the Value Chain—the Case of Hungary. *Sustainability*, 10(10):3491, 2018.

[3] Xifan Yao, Jiajun Zhou, Jiangming Zhang, and Claudio R Boër. From Intelligent Manufacturing to Smart Manufacturing for Industry 4.0 Driven by Next Generation Artificial Intelligence and Further On. In *ES*, pages 311–318. IEEE, 2017.

[4] David R Sjödin, Vinit Parida, Markus Leksell, and Aleksandar Petrovic. Smart Factory Implementation and Process Innovation: A Preliminary Maturity Model for Leveraging Digitalization in Manufacturing Moving to Smart Factories Presents Specific Challenges That Can be Addressed Through a Structured Approach Focused on People, Processes, and Technologies. *Research-technology management*, 61(5):22–31, 2018.

[5] Yoram Koren, Uwe Heisel, Francesco Jovane, Toshimichi Moriwaki, Gumter Pritschow, Galip Ulsoy, and Hendrik Van Brussel. Reconfigurable Manufacturing Systems. *CIRP annals*, 48(2):527–540, 1999.

[6] Jillian MacBryde, Tim Reckordt, Remi Zante, and Benoit Fernandez. The Impact of Covid on UK Manufacturing Firms and Supply Chains:[Stage 1 Report]. 2021.

[7] Yan Lu, Katherine C Morris, and Simon Frechette. Current Standards Landscape for Smart Manufacturing Systems. *National Institute of Standards and Technology, NISTIR*, 8107, 2016.

[8] Ray Y Zhong, Xun Xu, Eberhard Klotz, and Stephen T Newman. Intelligent Manufacturing in the Context of Industry 4.0: A Review. *Engineering*, 3(5):616–630, 2017.

[9] Jeff Morgan, Mark Halton, Yuansong Qiao, and John G Breslin. Industry 4.0 Smart Reconfigurable Manufacturing Machines. *Journal of Manufacturing Systems*, 59:481–506, 2021.

[10] Yoram Koren and Moshe Shpitalni. Design of Reconfigurable Manufacturing Systems. *Journal of Manufacturing Systems*, 29(4):130–141, 2010.

[11] Yoram Koren, Xi Gu, and Weihong Guo. Reconfigurable Manufacturing Systems: Principles, Design, and Future Trends. *Frontiers of Mechanical Engineering*, 13:121–136, 2018.

[12] Simok Lee, Sang-Hyuk Byun, Choong Yeon Kim, Sungwoo Cho, Steve Park, Joo Yong Sim, and Jae-Woong Jeong. Beyond Human Touch Perception: An Adaptive Robotic Skin Based on Gallium Microgranules for Pressure Sensory Augmentation. *Advanced Materials*, 34(44):2204805, 2022.

[13] Antonio Giovannini, Alexis Aubry, Hervé Panetto, Michele Dassisti, and Hind El Haouzi. Ontology-Based System for Supporting Manufacturing Sustainability. *Annual Reviews in Control*, 36(2):309–317, 2012.

[14] Eeva Järvenpää, Niko Siltala, Otto Hylli, and Minna Lanz. The Development of an Ontology for Describing the Capabilities of Manufacturing Resources. *Journal of Intelligent Manufacturing*, 30(2):959–978, 2019.

[15] Xin Huang, Cecilia Zanni-Merk, and Bruno Crémilleux. Enhancing Deep Learning with Semantics: An Application to Manufacturing Time Series Analysis. *Procedia Computer Science*, 159:437–446, 2019.

[16] Qiushi Cao, Ahmed Samet, Cecilia Zanni-Merk, François de Bertrand de Beuvron, and Christoph Reich. Combining Chronicle Mining and Semantics for Predictive Maintenance in Manufacturing Processes. *Semantic Web*, 11(6):927–948, 2020.

[17] Nitesh Khilwani, Jennifer A Harding, and Alok K Choudhary. Semantic Web in Manufacturing. *Proceedings of the Institution of Mechanical Engineers, Part B: Journal of Engineering Manufacture*, 223(7):905–924, 2009.

[18] Luis Ramos. Semantic Web for Manufacturing, Trends and Open Issues: Toward a State of the Art. *Computers & Industrial Engineering*, 90:444–460, 2015.

[19] Michael Gruninger and Christopher Menzel. The Process Specification Language (PSL) Theory and Applications. *AI magazine*, 24(3):63–63, 2003.

[20] Steven R Ray and Albert T Jones. Manufacturing Interoperability. *Journal of Intelligent Manufacturing*, 17(6):681–688, 2006.

[21] Yuqian Lu, Qun Shao, Chirpreet Singh, Xun Xu, and Xinfeng Ye. Ontology for Manufacturing Resources in a Cloud Environment. *International Journal of Manufacturing Research*, 9(4):448–469, 2014.

[22] Tianri Wang, Shunsheng Guo, and Chi-Guhn Lee. Manufacturing Task semantic Modeling and Description in Cloud Manufacturing System. *The International Journal of Advanced Manufacturing Technology*, 71(9):2017–2031, 2014.

[23] Li Zhou, Jianfeng Gao, Di Li, and Heung-Yeung Shum. The Design and Implementation of Xiaoice, An Empathetic Social Chatbot. *Computational Linguistics*, 46(1):53–93, 2020.

[24] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic Parsing on Freebase from Question-Answer Pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, 2013.

[25] Hongwei Wang, Fuzheng Zhang, Xing Xie, and Minyi Guo. DKN: Deep Knowledge-Aware Network for News Recommendation. In *Proceedings of the 2018 World Wide Web Conference*, pages 1835–1844, 2018.

[26] Fuzheng Zhang, Nicholas Jing Yuan, Defu Lian, Xing Xie, and Wei-Ying Ma. Collaborative Knowledge Base Embedding for Recommender Systems. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 353–362, 2016.

[27] Vito Bellini, Vito Walter Anelli, Tommaso Di Noia, and Eugenio Di Sciascio. Auto-Encoding User Ratings via Knowledge Graphs in Recommendation Scenarios. In *Proceedings of the 2nd Workshop on Deep Learning for Recommender Systems*, pages 60–66, 2017.

[28] Jeffrey Dalton, Laura Dietz, and James Allan. Entity Query Feature Expansion Using Knowledge Base Links. In *Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 365–374, 2014.

[29] Hadas Raviv, Oren Kurland, and David Carmel. Document Retrieval Using Entity-Based Language Models. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 65–74, 2016.

[30] Faezeh Ensan and Ebrahim Bagheri. Document Retrieval Model Through Semantic Linking. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 181–190, 2017.

[31] Xiaojun Chen, Shengbin Jia, and Yang Xiang. A Review: Knowledge Reasoning over Knowledge Graph. *Expert Systems with Applications*, 141:112948, 2020.

[32] Xiaohan Zou. A Survey on Application of Knowledge Graph. In *Journal of Physics: Conference Series*, volume 1487, page 012016. IOP Publishing, 2020.

[33] Xiaolu Lu, Soumajit Pramanik, Rishiraj Saha Roy, Abdalghani Abujabal, Yafang Wang, and Gerhard Weikum. Answering Complex Questions by Joining Multi-Document Evidence with Quasi Knowledge Graphs. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 105–114, 2019.

[34] Bo Qiao, Kui Fang, Yiming Chen, and Xinghui Zhu. Building Thesaurus-Based Knowledge Graph Based on Schema Layer. *Cluster Computing*, 20:81–91, 2017.

[35] Danilo Dessì, Francesco Osborne, Diego Reforgiato Recupero, Davide Buscaldi, Enrico Motta, and Harald Sack. Ai-kg: An Automatically Generated Knowledge Graph of Artificial Intelligence. In *International Semantic Web Conference*, pages 127–143. Springer, 2020.

[36] Irlán Grangel-González, Lavdim Halilaj, Sören Auer, Steffen Lohmann, Christoph Lange, and Diego Collarana. An RDF-Based Approach for Implementing Industry 4.0 Components with Administration Shells. In *2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–8. IEEE, 2016.

[37] Bin Zhou, Jinsong Bao, Jie Li, Yuqian Lu, Tianyuan Liu, and Qiwan Zhang. A Novel Knowledge Graph-Based Optimization Approach for Resource Allocation in Discrete Manufacturing Workshops. *Robotics and Computer-Integrated Manufacturing*, 71:102160, 2021.

[38] Liqiao Xia, Pai Zheng, Xinyu Li, Robert X Gao, and Lihui Wang. Toward Cognitive Predictive Maintenance: A Survey of Graph-Based Approaches. *Journal of Manufacturing Systems*, 64:107–120, 2022.

[39] Georg Buchgeher, David Gabauer, Jorge Martinez-Gil, and Lisa Ehrlinger. Knowledge Graphs in Manufacturing and Production: A Systematic Literature Review. *IEEE Access*, 9:55537–55554, 2021.

[40] Agniva Banerjee, Raka Dalal, Sudip Mittal, and Karuna Pande Joshi. Generating Digital Twin Models Using Knowledge Graphs for Industrial Production Lines. *UMBC Information Systems Department*, 2017.

[41] Jingshu Yuan and Hongqi Li. Research on the Standardization Model of Data Semantics in the Knowledge Graph Construction of Oil & Gas Industry. *Computer Standards & Interfaces*, 84:103705, 2023.

[42] Justin J Miller. Graph Database Applications and Concepts with Neo4j. In *Proceedings of the southern association for information systems conference, Atlanta, GA, USA*, volume 2324, pages 141–147, 2013.

[43] Mark A Musen. The Protégé Project: A Look Back and Look Forward. *AI matters*, 1 (4):4–12, 2015.

[44] André Henrique Dantas Neves Cordeiro. Apache Jena. *Acedido em*, 2, 2019.

[45] Renzo Angles and Claudio Gutierrez. Survey of Graph Database Models. *ACM Computing Surveys (CSUR)*, 40(1):1–39, 2008.

[46] Karlis Cerans, Guntis Barzdins, Renars Liepins, Julija Ovcinnikova, Sergejs Rikacovs, and Arturs Sprogis. Graphical Schema Editing for Stardog OWL/RDF Databases using OWLGrEd/S. In *OWLED*, volume 849, 2012.

[47] Paulo Pinheiro, Henrique Santos, Zhicheng Liang, Yue Liu, Sabbir M Rashid, Deborah L McGuinness, and Marcello Peixoto Bax. HADatAc: A Framework for Scientific Data Integration using Ontologies. In *ISWC (P&D/Industry/BlueSky)*, 2018.

[48] Dan Song, Xin Ye, Wenrong Wu, Zhijing Zhang, and Min Sheng. Ontology-Based Assembly Knowledge Representation and Process File Generation. In *IIMSS*, pages 95–105, 2022.

[49] Ihssen Belhadj and Thierry Boudemaghe. Semantic Querying of Hospital Data Using an Ontology-Based Model of Discharge Summaries and ICD 10. In *MIE*, pages 417–421, 2020.

[50] Audrey Berquand, Francesco Murdaca, Annalisa Riccardi, Tiago Soares, Sam Generé, Norbert Brauer, and Kartik Kumar. Artificial Intelligence for the Early Design Phases of Space Missions. In *2019 IEEE Aerospace Conference*, pages 1–20. IEEE, 2019.

[51] Benedikt Perak. Developing the Ontological Model for Research and Representation of Commemoration Speeches in Croatia Using a Graph Property Database. *Digital Humanities: Empowering Visibility of Croatian Cultural Heritage; Cambridge University Press: Cambridge, UK*, pages 88–111, 2020.

[52] Yang Lu. Industry 4.0: A Survey on Technologies, Applications and Open Research Issues. *Journal of Industrial Information Integration*, 6:1–10, 2017.

[53] Omkarprasad S Vaidya and Sushil Kumar. Analytic Hierarchy Process: An Overview of Applications. *European Journal of operational research*, 169(1):1–29, 2006.

[54] Majid Behzadian, S Khanmohammadi Otaghsara, Morteza Yazdani, and Joshua Ignatius. A State-of the-Art Survey of TOPSIS Applications. *Expert Systems with Applications*, 39(17):13051–13069, 2012.

[55] B Kadir, Y Zhang, and MJ Smith. Robotic Applications in Smart Manufacturing: A Review. *Robotics and Computer-Integrated Manufacturing*, 44:199–210, 2017.

[56] Boyi Li, Bingsheng Hou, Wenwu Yu, Xue Lu, and Chun Yang. Smart Manufacturing Systems for Industry 4.0: Conceptual Framework, Scenarios, and Future Perspectives. *Frontiers of Mechanical Engineering*, 13(2):137–150, 2018.

[57] Zhuming Bi, Shengyao Lang, Weiming Shen, and Li Wang. Reconfigurable Manufacturing Systems: Principles, Design, and Future Trends. *Frontiers of Mechanical Engineering*, 14(2):121–136, 2019.

[58] Gleb Belov, Tobias Czauderna, Maria Garcia de la Banda, Matthias Klapperstueck, Ilankaikone Senthooran, Mitch Smith, Michael Wybrow, and Mark Wallace. Process Plant Layout Optimization: Equipment Allocation. In *Principles and Practice of Constraint Programming: 24th International Conference, CP 2018, Lille, France, August 27-31, 2018, Proceedings 24*, pages 473–489. Springer, 2018.

[59] Gediminas Adomavicius and YoungOk Kwon. Improving Aggregate Recommendation Diversity Using Ranking-Based Techniques. *IEEE Transactions on Knowledge and Data Engineering*, 24(5):896–911, 2011.

[60] Antonio C Caputo, Pacifico M Pelagagge, Mario Palumbo, and Paolo Salini. Safety-Based Process Plant Layout Using Genetic Algorithm. *Journal of Loss Prevention in the Process Industries*, 34:139–150, 2015.

[61] Matthias Klar, Pascal Langlotz, and Jan C Aurich. A Framework for Automated Multi-objective Factory Layout Planning Using Reinforcement Learning. *Procedia CIRP*, 112: 555–560, 2022.

[62] Abdelghani Souilah. Simulated Annealing for Manufacturing Systems Layout Design. *European Journal of Operational Research*, 82(3):592–614, 1995.

[63] Thorsten Wuest, Daniel Weimer, Christopher Irgens, and Klaus-Dieter Thoben. Machine Learning in Manufacturing: Advantages, Challenges, and Applications. *Production & Manufacturing Research*, 4(1):23–45, 2016.

[64] Wenwen Li, Ender Özcan, and Robert John. Multi-Objective Evolutionary Algorithms and Hyper-Heuristics for Wind Farm Layout Optimisation. *Renewable Energy*, 105:473–482, 2017.

[65] Yang Liu, Shuangqing Chen, Bing Guan, and Ping Xu. Layout Optimization of Large-Scale Oil–Gas Gathering System Based on Combined Optimization Strategy. *Neurocomputing*, 332:159–183, 2019.

[66] Hichem Haddou Benderbal and Lyes Benyoucef. Machine Layout Design Problem under Product Family Evolution in Reconfigurable Manufacturing Environment: A Two-Phase-Based AMOSA Approach. *The International Journal of Advanced Manufacturing Technology*, 104:375–389, 2019.

[67] S Ghanei and T AlGeddawy. An Integrated Multi-Period Layout Planning and Scheduling Model for Sustainable Reconfigurable Manufacturing Systems. *Journal of Advanced Manufacturing Systems*, 19(01):31–64, 2020.

[68] Abdelkrim R Yelles-Chaouche, Evgeny Gurevsky, Nadjib Brahimi, and Alexandre Dolgui. Reconfigurable Manufacturing Systems from an Optimisation Perspective: A Focused Review of Literature. *International Journal of Production Research*, 59(21):6400–6418, 2021.

[69] Kaishu Xia, Christopher Sacco, Max Kirkpatrick, Clint Saidy, Lam Nguyen, Anil Kircaliali, and Ramy Harik. A Digital Twin to Train Deep Reinforcement Learning Agent for Smart Manufacturing Plants: Environment, Interfaces and Intelligence. *Journal of Manufacturing Systems*, 58:210–230, 2021.

[70] Peter Trebuňa, Miriam Pekarčíková, and Marián Petrik. Application of Tecnomatix Process Simulate for Optimisation of Logistics Flows. *Acta Montanistica Slovaca*, 23(4), 2018.

[71] Christine Connolly. Technology and Applications of ABB RobotStudio. *Industrial Robot: An International Journal*, 36(6):540–545, 2009.

[72] Radovan Holubek, Daynier Rolando Delgado Sobrino, Peter Košt'ál, and Roman Ružarovskỳ. Offline Programming of an ABB Robot Using Imported CAD Models in the RobotStudio Software Environment. *Applied Mechanics and Materials*, 693:62–67, 2014.

[73] Jarrod A COLETTA and Vedang CHAUHAN. Teaching Industrial Robot Programming using FANUC ROBOGUIDE and iRVision Software. In *16th International Multi-Conference on Society, Cybernetics and Informatics, IMSCI 2022*, pages 45–50, 2022.

[74] Fusaomi Nagata, Yudai Okada, Takamasa Kusano, and Keigo Watanabe. CLS Data Interpolation along Spline Curves and Post Processing for FANUC Industrial Robots. *Journal of Institute of Industrial Applications Engineers*, 5(3):129–135, 2017.

[75] Sudip Chakraborty and PS Aithal. ABB IRB 120-30.6 Build Procedure in RoboDK. *International Journal of Management, Technology and Social Sciences (IJMTS)*, 6(2): 256–264, 2021.

[76] Michal Hovanec, Peter Korba, and M Solc. Tecnomatix for Successful Application in the Area of Simulation Manufacturing and Ergonomics. *Proceedings of the Informatics, Geoinformatics and Remote Sensing*, 1:347–352, 2015.

[77] Pedro Neto. A guide for abb robot studio, 2014.

[78] Vladimír Bulej, Michal Bartoš, Vladimír Tlach, Martin Bohušík, and Dariusz Wiecek. Simulation of Manipulation Task Using iRVision Aided Robot Control in Fanuc RoboGuide Software. In *IOP Conference Series: Materials Science and Engineering*, volume 1199, page 012091. IOP Publishing, 2021.

[79] Sudip Chakraborty and PS Aithal. Forward and Inverse Kinematics Demonstration Using RoboDK and C. *International Journal of Applied Engineering and Management Letters (IJAEML)*, 5(1):97–105, 2021.

[80] Jan Baumgärtner, Malte Hansjosten, Dominik Schönhofen, and Ing Jürgen Fleischer. PyBullet Industrial: A Process-Aware Robot Simulation. *Journal of Open Source Software*, 8(85):5174, 2023.

[81] Jacques Denavit and Richard S Hartenberg. A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices. *Journal of Applied Mechanics*, 1955.

[82] Christoph Klug, Dieter Schmalstieg, Thomas Gloor, and Clemens Arth. A Complete Workflow for Automatic Forward Kinematics Model Extraction of Robotic Total Stations Using the Denavit-Hartenberg Convention. *Journal of Intelligent & Robotic Systems*, 95: 311–329, 2019.

[83] Megan Flanders and Richard C Kavanagh. Build-A-Robot: Using Virtual Reality to Visualize the Denavit–Hartenberg Parameters. *Computer Applications in Engineering Education*, 23(6):846–853, 2015.

[84] SHEN Luyang and PENG Yichao. Study on Trajectory Optimization Algorithm of Industrial Robot in Joint Space. In *Journal of Physics: Conference Series*, volume 1676, page 012206. IOP Publishing, 2020.

[85] Kalyanmoy Deb and Himanshu Gupta. Searching for Robust Pareto-Optimal Solutions in Multi-Objective Optimization. In *Evolutionary Multi-Criterion Optimization: Third International Conference, EMO 2005, Guanajuato, Mexico, March 9-11, 2005. Proceedings 3*, pages 150–164. Springer, 2005.

[86] Jürgen Teich. Pareto-Front Exploration with Uncertain Objectives. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 314–328. Springer, 2001.

[87] Shubham Jain and Vishal Parashar. Comparison of Priori and Posteriori Approach of Multi-Objective Optimization for WEDM on Ti6Al4V Alloy. *Materials Research Express*, 9(7):076504, 2022.

[88] Ch Achillas, D Aidonis, Ch Vlachokostas, N Moussiopoulos, G Banias, and D Triantafillou. A Multi-Objective Decision-Making Model to Select Waste Electrical and Electronic Equipment Transportation Media. *Resources, Conservation and Recycling*, 66:76–84, 2012.

[89] Feng Li, Lin Zhang, T Warren Liao, and Yongkui Liu. Multi-Objective Optimisation of Multi-Task Scheduling in Cloud Manufacturing. *International Journal of Production Research*, 57(12):3847–3863, 2019.

[90] AWA Hammad, A Akbarnezhad, and D Rey. A Multi-Objective Mixed Integer Nonlinear Programming Model for Construction Site Layout Planning to Minimise Noise Pollution and Transport Costs. *Automation in Construction*, 61:73–85, 2016.

[91] Pablo Pérez-Gosende, Josefa Mula, and Manuel Díaz-Madroñero. A Bottom-Up Multi-Objective Optimisation Approach to Dynamic Facility Layout Planning. *International Journal of Production Research*, pages 1–18, 2023.

[92] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[93] Eckart Zitzler and Lothar Thiele. Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, 1999.

[94] Carlos A Coello Coello. *Evolutionary algorithms for solving multi-objective problems*. Springer, 2007.

[95] Simon Huband, Philip Hingston, Luigi Barone, and Lyndon While. A Review of Multiobjective Test Problems and a Scalable Test Problem Toolkit. *IEEE Transactions on Evolutionary Computation*, 10(5):477–506, 2006.

[96] Xiaojie Tan, Kai Zheng, Yuhai Zhang, and Xia Yuan. Research on the Construction of a Smart City Transportation System Knowledge Graph. *Journal of Ambient Intelligence and Humanized Computing*, 2021.

[97] Xuejie Hao, Zheng Ji, Xiuhong Li, Lizeyan Yin, Lu Liu, Meiying Sun, Qiang Liu, and Rongjin Yang. Construction and Application of a Knowledge Graph. *Remote Sensing*, 13(13):2511, 2021.

[98] Mei Yong Zhang and Rong Zhi Du. A Real-time Inference Method of Graph Attention Network Based on Knowledge Graph for Lung Cancer. In *IWOST-2*, pages 192–199. Springer, 2021.

[99] Jolly Shah, SS Rattan, and BC Nakra. End-Effector Position Analysis Using Forward Kinematics for 5 DOF Pravak Robot Arm. *International Journal of Robotics and Automation*, 2(3):112–116, 2013.

[100] Vikas Yadav and Steven Bethard. A Survey on Recent Advances in Named Entity Recognition from Deep Learning Models. *arXiv preprint arXiv:1910.11470*, 2019.

[101] Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen-tau Yih. Cross-Sentence n-ary Relation Extraction with Graph LSTMs. *Transactions of the Association for Computational Linguistics*, 5:101–115, 2017.

[102] Lin Sun. Research on Product Attribute Extraction and Classification Method for Online Review. In *2017 International Conference on Industrial Informatics-Computing Technology, Intelligent Technology, Industrial Information Integration (ICIICII)*, pages 117–121. IEEE, 2017.

[103] Soonjo Kwon, Laetitia V Monnier, Raphael Barbau, and William Z Bernstein. Enriching Standards-based Digital Thread by Fusing As-designed and As-inspected Data Using Knowledge Graphs. *Advanced Engineering Informatics*, 46:101102, 2020.

[104] Zhe Chen, Yuehan Wang, Bin Zhao, Jing Cheng, Xin Zhao, and Zongtao Duan. Knowledge Graph Completion: A Review. *IEEE Access*, 8:192435–192456, 2020.

[105] Haibo Liu, Guoyi Jiang, Linhua Su, Yang Cao, Fengxin Diao, and Lipeng Mi. Construction of Power Projects Knowledge Graph Based on Graph Database Neo4j. In *CITS*, pages 1–4. IEEE, 2020.

[106] Nyoman Gunantara. A Review of Multi-objective Optimization: Methods and Its Applications. *Cogent Engineering*, 5(1):1502242, 2018.

[107] Sanyapong Petchrompo, Anupong Wannakrairot, and Ajith Kumar Parlikad. Pruning Pareto Optimal Solutions for Multi-objective Portfolio Asset Management. *European Journal of Operational Research*, 297(1):203–220, 2022.

[108] Niannian Guan, Dandan Song, and Lejian Liao. Knowledge Graph Embedding with Concepts. *Knowledge-Based Systems*, 164:38–44, 2019.

[109] Changlong Zhao, Chen Ma, Haifeng Zhang, Zhenrong Ma, Junbao Yang, Ming Li, Xuxu Wang, and Qiyin Lv. Modeling Manufacturing Resources Based on Manufacturability Features. *Scientific Reports*, 12(1):1–12, 2022.

[110] Wangwei Chu, Yingguang Li, Changqing Liu, Wenping Mou, and Limin Tang. A Manufacturing Resource Allocation Method with Knowledge-Based Fuzzy Comprehensive Evaluation for Aircraft Structural Parts. *International Journal of Production Research*, 52(11):3239–3258, 2014.

[111] Zeshui Xu and Qingli Da. An Approach to Improving Consistency of Fuzzy Preference Matrix. *Fuzzy Optimization and Decision Making*, 2(1):3–12, 2003.

[112] Zhi-Hong Zou, Yun Yi, and Jing-Nan Sun. Entropy Method for Determination of Weight of Evaluating Indicators in Fuzzy Synthetic Evaluation for Water Quality Assessment. *Journal of Environmental Sciences*, 18(5):1020–1023, 2006.

[113] Y-Y Hsu and K-L Ho. Fuzzy Expert Systems: An Application to Short-Term Load Forecasting. In *IEE Proceedings C (Generation, Transmission and Distribution)*, volume 139, pages 471–477. IET, 1992.

[114] Mu-Song Chen and Shinn-Wen Wang. Fuzzy Clustering Analysis for Optimizing Fuzzy Membership Functions. *Fuzzy Sets and Systems*, 103(2):239–254, 1999.