



**An On-Demand Fixture Manufacturing Cell for Mass
Customisation Production Systems**

Submitted By:

Enrico Naidoo (BScEng, UKZN) – 212539072

Supervisor:

Dr. Jared Padayachee

Co-Supervisor:

Prof. Glen Bright

December 2017

Submitted in fulfilment of the degree of:
Master of Science in Engineering (Mechanical Engineering);
College of Agriculture, Engineering and Science; University of KwaZulu-Natal.

Declaration: Submission and Originality

Supervisor

As the Candidate's Supervisor, I agree to the submission of this dissertation.

Signed:

Date:

Dr. Jared Padayachee

Co-Supervisor

As the Candidate's Co-Supervisor, I agree to the submission of this dissertation.

Signed:

Date:

Prof. Glen Bright

Student

I declare this dissertation has not previously been submitted for any degree in this, or any other university, and it is the student's (MSc Eng Candidate Enrico Naidoo's) own original work.

Signed:

Date:

Mr. Enrico Naidoo

Declaration: Plagiarism

I, Enrico Naidoo (212539072), declare that:

1. The research reported in this dissertation, except where otherwise indicated, is my original research.
2. This dissertation has not been submitted for any degree or examination at any other university.
3. This dissertation does not contain other persons' data, pictures, graphs or other information, unless specifically acknowledged as being sourced from other persons.
4. This dissertation does not contain other persons' writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:
 - a. Their words have been re-written but the general information attributed to them has been referenced
 - b. Where their exact words have been used, then their writing has been placed in italics and inside quotation marks, and referenced.
5. This dissertation does not contain text, graphics or tables copied and pasted from the internet, unless specifically acknowledged, and the source being detailed in the dissertation and in the References section.

Signed:

Date:

Mr. Enrico Naidoo

Declaration: Publications

Details of contribution to peer-reviewed publications that include research presented in this dissertation. The undersigned agree that the following submissions were made and accepted as described and that the content therein is contained in this research.

Published: ICINCO 2017

E. Naidoo, J. Padayachee, and G. Bright, “Optimal Scheduling of an on-Demand Fixture Manufacturing Cell for Mass Customisation Production Systems - Model Formulation, Presentation and Validation,” in Proceedings of the 14th International Conference on Informatics in Control, Automation and Robotics (ICINCO), 2017, vol. 1, pp. 17–24 (8 pages).

The paper was published and presented on 26 July 2017 in Madrid, Spain.

Enrico Naidoo was the lead author for this paper and conducted all research and experimentation under the supervision of the other two authors.

Accepted: IEEE ICCA 2018

E. Naidoo, J. Padayachee, G. Bright, “Scheduling Technique for Customised Parts with Modular Fixtures in On-Demand Fixture Manufacturing Cells,” in Proceedings of the 14th IEEE International Conference on Control and Automation (ICCA), 2018, vol. TBD, pp. TBD (6 pages).

The paper was accepted for publication on 13 March 2018, to be presented on 12-15 June 2018 in Alaska, USA.

Enrico Naidoo was the lead author for this paper and conducted all research and experimentation under the supervision of the other two authors.

Accepted: LNEE

E. Naidoo, J. Padayachee, G. Bright, “Scheduling of an On-Demand Fixture Manufacturing Cell for Mass Customization: Optimal Method vs. Heuristic,” in Lecture Notes in Electrical Engineering (LNEE), New York: Springer, TBD, pp. TBD (20 pages).

The chapter was accepted on 30 August 2017 as an extension of the ICINCO paper; to be published in a forthcoming entry of the Springer LNEE series.

Enrico Naidoo was the lead author for this chapter and conducted all research and experimentation under the supervision of the other two authors.

Submitted: SAJIE

E. Naidoo, J. Padayachee, G. Bright, “A Multi-Stage Optimisation Method for the Management of an On-Demand Fixture Manufacturing Cell for Mass Customisation Production Systems”, South African Journal of Industrial Engineering (SAJIE), vol. TBD, pp. TDB (13 Pages).

The paper was submitted on 23 August 2017 and is awaiting feedback.

Enrico Naidoo was the lead author for this paper and conducted all research and experimentation under the supervision of the other two authors.

Signed:

Date:

Mr. Enrico Naidoo

Acknowledgements

First and foremost, I would like to thank God for giving me the courage to pursue this research, and the ability to complete it.

I would like to thank my supervisor, Dr. Jared Padayachee, for his supervision, guidance and mentorship during the entirety of this research. He went beyond what was expected of him, for which I am very grateful.

I would like to thank my co-supervisor, Prof. Glen Bright, for providing the opportunity to be part of his Mechatronics and Robotics Research Group (MR2G), the constructive feedback I received from him for my publications, and the confident presentation of our conference paper in Spain.

I would like to acknowledge the National Research Foundation (NRF) and the University of KwaZulu-Natal (UKZN); the NRF for providing financial support for the research through the Blue Sky Research Grant (91339) and financial support for myself through the Freestanding, Scarce Skills and Innovation Scholarship for Masters Studies over the duration of 2016 and 2017; and UKZN for the remission of fees for 2016 as a first-year Masters student.

I would like to thank Dr. Anthony John Walker of UKZN, for his advice and insights on the context of the research topic.

I would like to thank my family for their support, encouragement and assistance throughout the trials and tribulations of the research.

Lastly, I would like to thank Ms Darshni Appalsamy for her efficient assistance with compiling the dissertation appendices in the required format.

Abstract

Increased demand for customised products has given rise to the research of mass customisation production systems. Customised products exhibit geometric differences that render the use of standard fixtures impractical. Fixtures must be configured or custom-manufactured according to the unique requirements of each product. Reconfigurable modular fixtures have emerged as a cost-effective solution to this problem. Customised fixtures must be made available to a mass customisation production system as rapidly as parts are manufactured. Scheduling the creation/modification of these fixtures must now be treated together with the production scheduling of parts on machines.

Scheduling and optimisation of such a problem in this context was found to be a unique avenue of research. An on-demand Fixture Manufacturing Cell (FxMC) that resides within a mass customisation production system was developed. This allowed fixtures to be created or reconfigured on-demand in a cellular manufacturing environment, according to the scheduling of the customised parts to be processed. The concept required the research and development of such a cell, together with the optimisation modelling and simulation of this cell in an appropriate manufacturing environment.

The research included the conceptualisation of a fixture manufacturing cell in a mass customisation production system. A proof-of-concept of the cell was assembled and automated in the laboratory. A three-stage optimisation method was developed to model and optimise the scheduling of the cell in the manufacturing environment. This included clustering of parts to fixtures; optimal scheduling of those parts on those fixtures; and a Mixed Integer Linear Programming (MILP) model to optimally synchronise the fixture manufacturing cell with the part processing cell. A heuristic was developed to solve the MILP problem much faster and for much larger problem sizes – producing good, feasible solutions. These problems were modelled and tested in MATLAB®. The cell was simulated and tested in AnyLogic®.

The research topic is beneficial to mass customisation production systems, where the use of reconfigurable modular fixtures in the manufacturing process cannot be optimised with conventional scheduling approaches. The results showed that the model optimally minimised the total idle time of the production schedule; the heuristic also provided good, feasible solutions to those problems. The concept of the on-demand fixture manufacturing cell was found to be capable of facilitating the manufacture of customised products.

Keywords: Mass customisation; Reconfigurable jigs and fixtures; Cellular manufacturing; Scheduling and optimisation

Contents

1	Introduction.....	1
1.1	Introduction.....	1
1.2	Background and Motivation.....	1
1.3	Existing Research and Research Gap.....	2
1.4	Aims and Objectives	3
1.5	Methodology	3
1.6	Research Contribution.....	4
1.7	Dissertation Overview.....	4
1.8	Summary	5
2	Literature Review.....	7
2.1	Introduction.....	7
2.2	Mass Customisation	7
2.3	Jigs and Fixtures	8
2.4	Reconfigurable Manufacturing Systems	13
2.5	Group Technology and Cellular Manufacturing	15
2.6	Production Planning.....	16
2.6.1	Push Production Systems	17
2.6.2	Push Production Systems	18
2.7	Optimisation Methods.....	20
2.7.1	Formulation.....	20
2.7.2	Solution Techniques.....	21
2.8	Scheduling.....	23
2.8.1	Job Shop Scheduling Problem	24
2.8.2	Research Studies	26
2.9	Additive Manufacturing.....	28

2.10	Summary	29
3	The Fixture Manufacturing Cell Concept	31
3.1	Introduction.....	31
3.2	The Concept.....	31
3.3	Test Product	32
3.4	Fixture Design.....	32
3.5	FxMC Layout.....	35
3.6	Production System Representation	37
3.7	Summary	39
4	Practical Implementation	41
4.1	Introduction.....	41
4.2	Existing Infrastructure and Modifications	41
4.3	Conveyor System Electronics	45
4.4	Pneumatic System.....	49
4.5	PLC	53
4.5.1	PLC Components	53
4.5.2	PLC Code (Ladder Logic Programme)	57
4.6	Summary	64
5	The Fixture Manufacturing Cell in Context of the Production Planning and Control System	65
5.1	Introduction.....	65
5.2	Push/Pull Combination	65
5.3	Production Planning and Control Schematic	65
5.4	Product Design.....	66
5.4.1	Digital Rendering of Customer Order.....	67
5.4.2	Fixture Configuration.....	67
5.4.3	Outputs of Product Design.....	67
5.5	Inventory Management	67

5.5.1	FxMC Inventory.....	68
5.5.2	FxMC Storage.....	68
5.5.3	PMS Inventory.....	68
5.6	Manufacturing Resource Planning (MRP II).....	68
5.6.1	Inputs to MRP II.....	69
5.6.2	Subsystems of MRP II.....	70
5.6.3	Outputs of MRP II.....	70
5.7	Shop Floor Control (SFC).....	70
5.7.1	FxMC Decisions.....	71
5.7.2	Other Functions.....	71
5.8	Summary.....	72
6	Scheduling Method.....	73
6.1	Introduction.....	73
6.2	Optimisation Requirements.....	73
6.3	Formulation.....	73
6.4	Method Overview.....	75
6.5	Method Notation.....	75
6.6	Method Assumptions.....	77
6.7	Clustering (Stage I).....	77
6.8	Intracluster Sequencing (Stage II).....	84
6.9	Final Scheduling (Stage III).....	86
6.9.1	MILP Model (Non-Linearised).....	86
6.9.2	MILP Model Description.....	87
6.9.3	Solution Technique.....	89
6.10	Stage III Heuristic.....	91
6.11	Summary.....	94
7	Agent-Based Simulation.....	95

7.1	Introduction.....	95
7.2	Simulation FxMC Layout	95
7.3	Agents and Resources	98
7.4	Process Modelling Flowchart.....	99
7.4.1	Fixture Flow I	101
7.4.2	Part Flow I.....	102
7.4.3	Match	102
7.4.4	Fixture Flow II	103
7.4.5	Part Flow II	103
7.5	Simulation Interface.....	104
7.6	Summary	105
8	Sample Problem.....	107
8.1	Introduction.....	107
8.2	The Problem.....	107
8.3	Stage I	108
8.4	Stage II.....	110
8.5	Stage III.....	112
8.6	Stage III Heuristic	113
8.7	Simulation.....	114
8.8	Summary	116
9	Testing and Results	117
9.1	Introduction.....	117
9.2	MDS Robustness.....	117
9.2.1	Aim	117
9.2.2	Methodology	117
9.2.3	Results.....	117
9.2.4	Analysis.....	118

9.2.5	Conclusion	119
9.3	Silhouette Values (Constant Parts, Variable Fixtures).....	119
9.3.1	Aim	119
9.3.2	Methodology	119
9.3.3	Results.....	119
9.3.4	Analysis.....	121
9.3.5	Conclusion	121
9.4	Cluster Reordering Effectiveness.....	122
9.4.1	Aim	122
9.4.2	Methodology	122
9.4.3	Results.....	122
9.4.4	Analysis.....	124
9.4.5	Conclusion	124
9.5	MILP Model Behaviour	125
9.5.1	Aim	125
9.5.2	Methodology	125
9.5.3	Results.....	125
9.5.4	Analysis.....	128
9.5.5	Conclusion	130
9.6	Heuristic Solution Quality	130
9.6.1	Aim	130
9.6.2	Methodology	130
9.6.3	Results.....	131
9.6.4	Analysis.....	133
9.6.5	Conclusion	134
9.7	Simulation Behaviour	134
9.7.1	Aim	134

9.7.2	Methodology	134
9.7.3	Results.....	134
9.7.4	Analysis.....	137
9.7.5	Conclusion	139
9.8	Summary	140
10	Discussion	141
10.1	Introduction.....	141
10.2	Concept Overview and Justification	141
10.3	The FxMC.....	142
10.4	PPC System with FxMC	143
10.5	Scheduling Method	144
10.5.1	Stage I	145
10.5.2	Stage II.....	146
10.5.3	Stage III.....	147
10.5.4	Stage III Alternative.....	148
10.6	Simulation.....	149
10.7	Shortcomings and Scope for Improvement.....	150
10.8	The FxMC Implications	152
10.9	Summary	154
11	Conclusion	155
11.1	Introduction.....	155
11.2	Research Aims and Objectives Evaluation	155
11.3	Research Contribution.....	156
11.4	Future Work.....	157
11.5	Summary	157
	References.....	159
A.	Appendix A: Testing Results	167

A.1.	MDS Robustness.....	167
A.2.	Silhouette Test (Constant Parts, Variable Fixtures).....	169
A.2.1.	Silhouette Plots	174
A.3.	Cluster Reordering	185
A.3.1.	Dendrograms	191
A.4.	MILP Behaviour	200
A.4.1.	MILP Convergence Graphs	202
A.5.	Heuristic Solution Quality	212
A.6.	Simulation Behaviour	212
A.6.1.	Idle Time Graphs	212
A.6.2.	Utilisation Graphs	218
B.	Appendix B: MATLAB Code.....	225
B.1.	Clustering (Stage I and Stage II).....	225
B.2.	MILP Model (Stage III)	233
B.3.	Stage III Heuristic	242
C.	Appendix C: Standard Operating Procedures	245
C.1.	Fixture Fabrication.....	245
C.1.1.	Introduction.....	245
C.1.2.	Task Procedure.....	245
C.1.3.	Notes/Observations	245
C.2.	Fixture Reconfiguration	246
C.2.1.	Introduction.....	246
C.2.2.	Task Procedure.....	246
C.2.3.	Notes/Observations	246
C.3.	Part Processing.....	247
C.3.1.	Introduction.....	247
C.3.2.	Task Procedure.....	247

C.3.3. Notes/Observations	247
D. Appendix D: Catalogue Components	249
D.1. Mean Well® 12 V PSU	249
D.2. Huaguan Relays® 24 V DC Relay.....	249
D.3. Festo® Standard Cylinder.....	250
D.4. Festo® One-Way Flow Control Valve	251
D.5. Festo® Solenoid Valve	251
D.6. Festo® Silencer.....	252
D.7. Festo® Diffuse Light Sensor	252

List of Acronyms

2D	Two-Dimensional
3D	Three-Dimensional
AC	Alternating Current
ACO	Ant Colony Optimisation
AM	Additive Manufacturing
ASRS	Automated Storage & Retrieval System
ASTM	American Society for Testing and Materials
B&B	Branch and Bound
BDF	Biased Decision Feedback
BHN	Brinell Hardness Number
BOM	Bill of Materials
CAD	Computer Aided Design
CAFD	Computer Aided Fixture Design
CAM	Computer Aided Manufacturing
CAPP	Computer Aided Process Planning
CBR	Case-Based Reasoning
CM	Cellular Manufacturing
CNC	Computer Numerical Control
COM	Common
COMMS	Customer-Oriented Manufacturing Management System
COMS	Customer-Oriented Management System
CONWIP	Constant Work In Progress
CPU	Computer Processing Unit
DBFA	Discrete Bacteria
DC	Direct Current
DFM	Design For Manufacture
DML	Dedicated Manufacturing Line
DP	Decoupling Point (as defined in Section 2.2)

DP	Dynamic Programming (as defined in Section 2.7.2)
EDD	Earliest Due Date
ERP	Enterprise Resource Planning
F-BOM	Fixture BOM
FIFO	First-In, First-Out
FJSSP	Flexible Job Shop Schedule Problem
F-MPS	Fixture MPS
FMS	Flexible Manufacturing System
FRL	Filter Regulator Lubricator
FSSP	Flow Shop Schedule Problem
FxMC	Fixture Manufacturing Cell
GA	Genetic Algorithm
GT	Group Technology
GUI	Graphical User Interface
IP	Integer Programming
JIT	Just-In-Time
JSSP	Job Shop Scheduling Problem
LB	Lower Bound
LDR	Light Department Resistance
LP	Linear Programming
MC	Mass Customisation
MDS	Multi-Dimensional Scaling
MES	Manufacturing Execution System
MILP	Mixed Integer Linear Programming
MIP	Mixed Integer Programming
MLP	Machine Layout Problem
MPS	Master Production Schedule
MRP	Material Requirements Planning
MRP II	Manufacturing Resource Planning
NC	Normally Closed

NEMA	Natural Electrical Manufacturers Association
NN	Neural Network
NO	Normally Open
NP	Non-Deterministic Polynomial-Time
P-BOM	Product BOM
PC	Personal Computer
PCB	Printed Circuit Board
PDM	Product Data Management
PFA	Product Flow Analysis
PLA	Polylactic Acid
PLC	Programmable Logic Controller
P-MPS	Product MPS
PMS	Product Manufacturing System
POK	Production Order Kanban
Pp	Postponement
PPC	Production Planning & Control
PSO	Particle System Optimisation
PSU	Power Supply Unit
RAM	Random Access Memory
RFID	Radio Frequency Identification
RMS	Reconfigurable Manufacturing System
ROC	Rank-Order Clustering
ROP	Re-Order Point
S3H	Stage 3 Heuristic
SA	Simulated Annealing
SCM	Supply Chain Management
SD	Standard Deviation
SFC	Shop Floor Control
SOP	Standard Operating Procedure
SPDT	Single Pole - Double Throw

SPT	Shortest Processing Time
TS	Tabu Search
UB	Upper Bound
UKZN	University of KwaZulu-Natal
VMC	Virtual Manufacturing Cell
VRP	Vehicle Routing Problem
WIP	Work In Progress
WK	Withdrawal Kanban

List of Nomenclature

a	positive binary matches
b	positive-to-negative binary mismatches
c	negative-to-positive binary mismatches
D	dissimilarity value
d	negative binary matches
I	set of groups of fixture-part mappings
i, \hat{i}	index of group of fixture-part mappings
j, \hat{j}	index of fixture-part mappings in a group
k, \check{k}	time period
m	number of fixtures
n	number of parts
$O\bar{T}E$	ladder logic energise coil
p	part number
P	set of parts
q	fixture number
Q	set of fixtures
R^2	coefficient of determination
S	similarity value
Str	Kruskal's normalised stress
XIC	ladder logic closed switch
$X_{ijk}, X_{ij\check{k}}$	binary decision variable
XIO	ladder logic open switch
Y_{iijjkk}	slack variable for linearisation of product
Z_{iijjkk}	slack variable for linearisation of absolute value
ρ_{ij}	fixture reconfiguration time
τ_{ij}	part processing time

List of Figures

Figure 1.1: Traditional fixture management	2
Figure 1.2: Proposed fixture management	2
Figure 2.1: Pin-array type fixture [25]	9
Figure 2.2: Phase-change material type fixture [26]	10
Figure 2.3: IMAO Corporation® grid hole modular fixture [3]	10
Figure 2.4: Adaptive fixture vice [28]	11
Figure 2.5: Steps to fixture design [29]	12
Figure 2.6: Idealised Cost vs. Capacity graph for FMS, RMS and DML [31]	13
Figure 2.7: Push system schematic [46]	17
Figure 2.8: Single card kanban system [48]	18
Figure 2.9: Two-card Kanban system [48]	19
Figure 2.10: Branch and Bound tree representation [58]	21
Figure 2.11: Information flow diagram in a manufacturing system [74]	24
Figure 3.1: A general FxMC decision flowchart	31
Figure 3.2: Pin configuration for large triangle part (isometric view)	33
Figure 3.3: Large triangle part on fixture (isometric view)	33
Figure 3.4: FxMC layout schematic	36
Figure 3.5: Workflow of production system representation	38
Figure 4.1: Lab FxMC	41
Figure 4.2: Automated Storage and Retrieval System used	42
Figure 4.3: ASRS Graphical User Interface	43
Figure 4.4: CNC router for fixture fabrication	43
Figure 4.5: Conveyors	44
Figure 4.6: Pallet with embedded RFID tag	45
Figure 4.7: Relay terminal connections	46

Figure 4.8: Soldered relay circuit.....	46
Figure 4.9: Soldered relay circuit underside with short-circuit breaks	47
Figure 4.10: Relay circuit with wire connections	47
Figure 4.11: Relay circuit housing being 3D-printed.....	48
Figure 4.12: PSU (left) and relay circuit in housing (right).....	48
Figure 4.13: Pneumatic circuit.....	49
Figure 4.14: FRL service unit	50
Figure 4.15: 5/2 Solenoid valve	51
Figure 4.16: Electrical circuit diagram for pneumatic system	51
Figure 4.17: Double-acting actuator and 5/2 control valve setup	52
Figure 4.18: Pneumatic components.....	53
Figure 4.19: PLC module with connectors	54
Figure 4.20: PLC connections schematic.....	54
Figure 4.21: Electrical circuit schematic for PLC output devices	55
Figure 4.22: Intersection point for fixtures to be reconfigured	56
Figure 4.23: PLC programme	58
Figure 4.24: PLC programme (initial state).....	59
Figure 4.25: Rung 0001 activation.....	59
Figure 4.26: Rung 0007 activation.....	59
Figure 4.27: Rung 0008 activation.....	60
Figure 4.28: Reconfiguration Mode ready	60
Figure 4.29: Sensor activated on Rung 0002	61
Figure 4.30: Valve activated on Rung 0005, waiting for timer to elapse	62
Figure 4.31: Motors 1 and 2 deactivated.....	62
Figure 4.32: Conveyor segment paused, pallet waiting at position	62
Figure 4.33: Piston rod extension to push pallet along branch conveyor	63
Figure 4.34: Waiting for piston rod to retract on Rung 0008 so procedure can be concluded	63

Figure 4.35: Sensor activated when Reconfiguration Mode is not activated.....	64
Figure 5.1: Schematic of PPC system, adapted from Groover [42].....	66
Figure 5.2: Graphical representation of reorder point system [42].....	68
Figure 5.3: Subsystems of MRP II system from Figure 5.1.....	70
Figure 5.4: Subsystems of Shop Floor Control system from Figure 5.1.....	71
Figure 6.1: Pin configuration for large square-shaped part.....	78
Figure 6.2: Binary matrix representation for Figure 6.1.....	78
Figure 6.3: Pin configuration to binary matrix representation for large triangular-shaped part.....	78
Figure 6.4: Elementwise relationships in practice.....	80
Figure 6.5: Sample problem MDS output.....	82
Figure 6.6: k-means Algorithm.....	82
Figure 6.7: k-means++ Algorithm.....	83
Figure 6.8: Sample problem k-means output. Data clustered to four clusters.....	83
Figure 6.9: Cluster Size Fail-Safe Algorithm.....	84
Figure 6.10: Reordering of Fixture 4 in sample problem.....	85
Figure 6.11: Branch and Bound Algorithm.....	90
Figure 6.12: Stage III Heuristic flowchart.....	92
Figure 6.13: Example of Stage III Heuristic actions on the transposed I matrix.....	93
Figure 6.14: Critical lag column (Column 3). Feasible solution unobtainable.....	93
Figure 7.1: Simulation FxMC layout, with Space Markup labels.....	95
Figure 7.2: Animation screenshot of cell layout.....	96
Figure 7.3: Fixture Fabrication Station.....	97
Figure 7.4: Fixture Reconfiguration Station.....	97
Figure 7.5: Part Processing Cell.....	98
Figure 7.6: Simulation Process Flowchart.....	100
Figure 7.7: Simulation GUI.....	105
Figure 8.1: Sample set of pin configurations.....	107

Figure 8.2: MDS plot (left) and k-means output (right).....	109
Figure 8.3: Silhouette plot for sample problem	110
Figure 8.4: Intracluster sequence for Fixture 1	110
Figure 8.5: Intracluster sequence for Fixture 2	111
Figure 8.6: Intracluster sequence for Fixture 3	111
Figure 8.7: Intracluster sequence for Fixture 4.....	111
Figure 8.8: Graph of convergence.....	112
Figure 8.9: Optimal schedule simulation	115
Figure 8.10: Heuristic schedule simulation.....	115
Figure 9.1: Average Silhouette Values for Increasing Fixture Quantity (Sample 1)	120
Figure 9.2: Average Silhouette Values for Increasing Fixture Quantity (Sample 2).....	120
Figure 9.3: Average Silhouette Values for Increasing Fixture Quantity (Sample 3).....	121
Figure 9.4: Average Improvement for Increasing Fixture Quantity	124
Figure 9.5: Variables Growth Characteristics for Increasing Fixtures and Constant Parts.....	126
Figure 9.6: Variables Growth Characteristics for Increasing Parts and Constant Fixtures.....	127
Figure 9.7: Nodes Explored in relation to Variables.....	127
Figure 9.8: Solution Time required in relation to Variables	128
Figure 9.9: Solution Time in comparison to Nodes Explored	128
Figure 9.10: Heuristic Solution discrepancies in comparison to Optimal Solutions	131
Figure 9.11: Discrepancy Increase in terms of Fixture Increase.....	132
Figure 9.12: Discrepancy in terms of Part Increase	132
Figure 9.13: Heuristic Solution Times in comparison to MILP Solution Times	133
Figure 9.14: cellFR Individual Idle Time overlay (100 parts, 5 fixtures).....	135
Figure 9.15: cellPP Individual Idle Time overlay (100 parts, 5 fixtures)	136
Figure 9.16: cellFR Individual Idle Times (400 parts, 40 fixtures)	136
Figure 9.17: cellPP Individual Idle Times (400 parts, 40 fixtures).....	137

List of Tables

Table 2.1: Requirements of fixture design [30]	12
Table 3.1: Fixture specifications	35
Table 6.1: Workflow table with alphabet representation	74
Table 6.2: Method notation sample explanations	76
Table 6.3: Sample problem non-metric distance matrix	80
Table 6.4: Workflow table with decision variable indices	89
Table 6.5: Workflow table with example indices	89
Table 6.6: MILP options used	90
Table 7.1: Flowchart block descriptions for Fixture Flow I	101
Table 7.2: Flowchart block descriptions for Fixture Flow II	102
Table 7.3: Flowchart block descriptions for Match	103
Table 7.4: Flowchart block descriptions for Fixture Flow II	103
Table 7.5: Flowchart block descriptions for Part Flow II	103
Table 8.1: Operation times	108
Table 8.2: Non-metric distance matrix	108
Table 8.3: Final schedule (optimal)	113
Table 8.4: Final schedule (heuristic)	114
Table 9.1: MDS results synthesis	118
Table 9.2: Silhouette values results synthesis	119
Table 9.3: Cluster reordering results synthesis	122
Table 9.4: Discrepancy results synthesis	131
Table 9.5: Simulation/Heuristic comparison results synthesis	135
Table 9.6: Utilisation results synthesis	137

1 Introduction

1.1 Introduction

This chapter introduces the research. The importance of mass customisation and reconfigurable fixtures are elaborated. A background of reconfigurable fixture management is presented. The on-demand fixture manufacturing cell concept is proposed and motivated. The gap in research is identified. The aims and objectives are stated, with the methodology presented thereafter. The research contributions are described. The dissertation overview is presented, with a summary of each chapter.

1.2 Background and Motivation

The fourth industrial revolution has placed an emphasis on customer-driven manufacturing at a large scale, such that product variety and economies of scale can both be achieved; the term to describe this manufacturing paradigm is Mass Customisation (MC) [1]. Mass customisation involves merging the benefits of both low volume and high volume production. Low volume production enables high product variety; however, throughput is reduced due to the non-standardisation of the activities involved. Conversely, high volume production enables high throughput due to the standardisation of activities, but the consequence is that of low product variety [2]. The realisation of mass customisation requires standardised procedures that are capable of adapting to a variety of products.

Reconfigurable fixtures are capable of facilitating mass customisation, through the adaptability provided by the concept. Fixtures are used to physically locate, hold and support parts during a manufacturing process [3]. Dedicated fixtures are prevalent in mass production, where the fixture design is applicable to a specific product design only. Customised products, however, can exhibit geometric differences that render the use of dedicated fixtures impractical. Fixtures should be configured or custom-manufactured for the unique requirements of each product in a MC production system [4]. Moreover, the traditional approach of outsourcing the production of fixtures to off-site facilities is not applicable to those of the reconfigurable type [5]. Reconfigurable fixtures should be made available to the system as rapidly as parts are manufactured. Scheduling the manufacture and dispatch of fixtures should be treated together with the production scheduling of parts on machines.

The research proposed the concept of an on-demand Fixture Manufacturing Cell (FxMC) which is to reside in a mass customisation production system. The FxMC is a specialised cell that deals with the manufacturing, storing, modifying and dispatching of the reconfigurable fixtures implemented in a MC production system. The FxMC serves the Product Manufacturing System (PMS), which fabricates parts for the customised products. The customised parts are expected to differ geometrically, based on customer demands. The fixtures should be supplied by the FxMC to the PMS in the appropriate configuration, where they are used to secure the customised parts during their respective manufacturing processes.

The traditional fixture management for mass production systems is represented in Figure 1.1. The fixtures are based on a single design, and are manufactured separately before production takes place. Product quantity forecasts dictate the number of fixtures fabricated. The same part design is served by

an unchanging fixture. The fixtures are interchangeable, so the sequence of parts (and thus fixtures) is inconsequential to the system performance. No intermediate step is required [5].

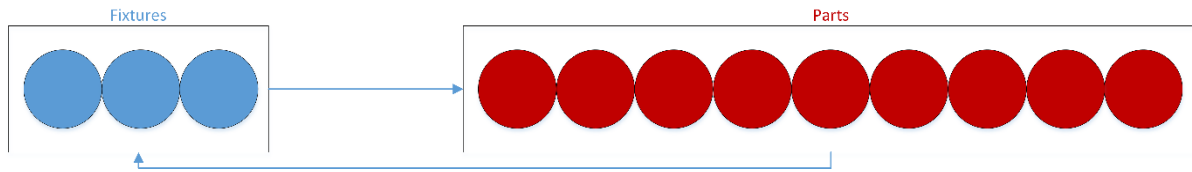


Figure 1.1: Traditional fixture management

The method described in Figure 1.1 is insufficient for implementation in a mass customisation system, since part types vary; thus, fixtures are not interchangeable. The proposed fixture management for MC systems is represented by Figure 1.2. Reconfigurable fixtures are implemented, which are to be fabricated in the FxMC. The fixtures must be appropriately reconfigured for the corresponding part type before being dispatched to that part. The performance of the production system is influenced by the sequencing and scheduling of both fixture reconfigurations and part processing with fixtures. The production planning problem that results from the FxMC concept created an avenue for optimisation that the research explored.

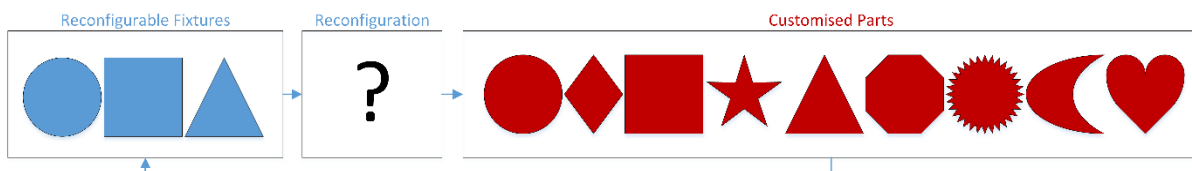


Figure 1.2: Proposed fixture management

1.3 Existing Research and Research Gap

The reviewed literature indicated that operations research in mass customisation was an area of interest [2]. Customer-driven manufacturing can provide a competitive advantage for manufacturing industries, which has driven the increased research activities in the field. Mass customisation requires alternative manufacturing systems to be employed, such as Flexible Manufacturing Systems (FMS), Reconfigurable Manufacturing Systems (RMS) and Cellular Manufacturing (CM) [6]. Optimisation techniques for scheduling activities in alternative manufacturing systems must be developed to provide the improvements in efficiency and best practices that similar endeavours have provided for mass production.

The surveyed scheduling studies revealed that fixture utilisation in a mass customisation system was predominantly limited to placing a constraint on the availability of fixtures as a resource [7]–[10]. Furthermore, the fixtures in these studies were of the standard, non-reconfigurable type. Optimisation research within the reconfigurable fixture research area was found to be limited to the fixture design itself [11]. Optimisation techniques for improving reconfigurable fixture design through Computer-Aided Fixture Design (CAFD) systems was found to be the research focus for reconfigurable fixtures. The research did not investigate a system that could manufacture and/or modify fixtures on-demand to satisfy the manufacturing system demands.

Bi and Zhang [4] stated (in 2001) that research of production planning techniques that enabled efficient utilisation of modular fixture (a type of reconfigurable fixture) components was yet to be investigated. Bi et al. [12] reiterated this claim in a future (2008) publication. The reviewed literature confirmed this research gap as current, despite the increased research interest in mass customisation and the capability of reconfigurable fixtures to facilitate MC.

The findings revealed a unique avenue for the research undertaken. The research problem was defined as: Can an on-demand fixture manufacturing cell, with suitable production planning and control, facilitate the manufacture of customised products?

The concept of an on-demand fixture manufacturing cell was explored to address reconfigurable fixture management in a mass customisation system. A proof-of-concept of the FxMC was developed and implemented as a conceptual solution. An optimisation model for the scheduling of the fixture manufacturing cell was developed to provide an operational framework for the FxMC.

1.4 Aims and Objectives

The aim of the research was to develop an on-demand fixture manufacturing cell and optimal scheduling method for mass customisation production systems.

The objectives were to:

- 1) Research, identify and implement suitable equipment to create a functional fixture manufacturing cell.
- 2) Research, select and integrate necessary electrical and electronic components for the automation of the cell.
- 3) Research, develop and implement software programmes for the control and automation of the mechanical, electrical and electronic systems of the cell.
- 4) Research and develop an optimisation model for the scheduling of the fixture manufacturing cell.
- 5) Research and develop a suitable simulation model for testing of the fixture manufacturing cell.
- 6) Test the performance of the cell, optimisation model and simulation against established manufacturing performance metrics.

1.5 Methodology

The aims and objectives were achieved by fulfilling the following methodology.

- 1) Performed a literature review on mass customisation and the production of reconfigurable fixtures.
- 2) Performed a literature review on production planning for fixture management and circulation.
- 3) Researched and identified suitable manufacturing technologies for the production of reconfigurable fixtures.
- 4) Designed a suitable manufacturing cell based on selected technologies; the cell was designed for the optimal flow of material and execution of activities.

- 5) Automated the cell through a combination of PC and PLC control.
- 6) Created a suitable optimisation model that determined the times, sequences and types of operations executed within the cell.
- 7) Simulated the performance of the model in MATLAB®.
- 8) Modified the optimisation model as required, based on simulation results.
- 9) Developed a simulation of the fixture manufacturing cell.
- 10) Tested the cell through the simulation and gathered performance data.
- 11) Validated the performance of the cell and model.
- 12) Documented research findings in a dissertation.

1.6 Research Contribution

The research findings contributed to the field of operations research in the mass customisation research area. The concept of an on-demand fixture manufacturing cell was developed for the management of reconfigurable fixtures in a production system. A cell layout for implementation was conceptualised. The cell was simulated with an agent-based simulation to verify its viability. An optimal scheduling method for the cell was developed and verified. A practicable heuristic was created to deal with larger schedules, which produced feasible, near-optimal solutions with minimal computational effort. The research also provided a production planning and control overview in which the fixture manufacturing cell could be implemented, and customised products could be integrated.

The optimal scheduling method comprised of three stages. The Clustering stage assigned parts to fixtures. The method used could be applied in isolation to define fixture-part mappings for reconfigurable modular fixtures in industry. The Intracluster Sequencing stage could be used to sequence the processing of the fixture-part mappings thereafter, if necessary. The Final Scheduling stage provided a model that optimised the operations in a two-cell manufacturing system. The model could be modified and applied to industries (outside of mass customisation) that could be represented by the system described, for the purpose of minimising total idle time. Thus, these techniques could also be applied separately, with some degree of modification, outside of the specific intended application in this research.

The outputs of the research could benefit manufacturing enterprises that aspire to gain a competitive advantage through the sale of customised products to consumers. Industries where mass customisation is an emerging competitive advantage include: automotive, electronics, clothing and appliance manufacturing industries [2].

1.7 Dissertation Overview

Chapter 1: Introduces the research and its necessity for the mass customisation field.

Chapter 2: A literature review of significant background topics related to the research.

Chapter 3: Introduces the manufacturing environment within which the research was implemented, and the subsystems used thereof, together with the fixture manufacturing cell layout.

Chapter 4: Documents the implementation of the laboratory fixture manufacturing cell, with descriptions of the subsystems employed.

Chapter 5: Describes the concept of the fixture manufacturing cell within the broader scope of the production planning and control system.

Chapter 6: Presents the three-stage optimal scheduling method developed for the fixture manufacturing cell, and the heuristic created to overcome the shortcomings of the third stage.

Chapter 7: Presents the agent-based simulation developed for large-scale testing of the fixture manufacturing cell presented in Chapter 4.

Chapter 8: Presents a sample problem that was solved and analysed using the scheduling method, the heuristic and the simulation.

Chapter 9: Documents the testing conducted on the various components of the scheduling method, the heuristic and the simulation.

Chapter 10: A discussion of the dissertation contents and research findings.

Chapter 11: A conclusion of the dissertation, with recommendations for future work.

1.8 Summary

The background and motivation of the on-demand fixture manufacturing cell concept was presented as a requirement for mass customisation and the reconfigurable fixtures implemented for customer-driven manufacturing. The research gap in scheduling and management of reconfigurable fixtures was established. The aims and objectives of the research were listed, before the methodology for achieving those goals was presented. The research contributions obtained were summarised, before an overview of the dissertation was listed.

2 Literature Review

2.1 Introduction

This chapter presents the significant background topics investigated for the research undertaken. The broad topics reviewed include: mass customisation; jigs and fixtures; manufacturing systems, such as the reconfigurable manufacturing system, and group technology with cellular manufacturing; production planning for manufacturing systems; mathematical optimisation methods; scheduling of manufacturing systems; and additive manufacturing as a technological facilitator of mass customisation.

2.2 Mass Customisation

The on-demand Fixture Manufacturing Cell (FxMC) was postulated as a facilitator for Mass Customisation (MC). The MC research field was investigated to suitably place the FxMC within the MC framework.

Mass customisation aims to produce individually customised products, with the volume, cost and efficiency of mass production. MC intends to improve overhead, price, profit and company success when compared to traditional job shops. However, the volume, cost and efficiency of MC is reduced when compared to traditional mass production [13]. Mass customisation research began in 1987, when companies turned to the concept to improve customer satisfaction, market share and company success. Customer integration techniques, modular design techniques, Flexible Manufacturing Systems (FMS) and Supply Chain Methods (SCM) were developed to help realise the idea of mass customisation [14].

The basic properties of MC are summarised as follows [15]:

- **Goal:** Providing cost-effective goods and services with sufficient customisation and variety.
- **Economics:** Customer integration and economies of scale.
- **Focus:** Customisation and variety via responsiveness and flexibility.
- **Product:** Customer-driven standardised modules and product families.
- **Key Features:**
 - unpredictable demand trends;
 - diverse niches;
 - high quality, low cost personalised goods and services;
 - minimal product development phases;
 - minimal product life cycles.
- **Organisation:** Adaptive and flexible.
- **Customer Involvement:** Customer configuration, co-design, and user innovation, amongst other such techniques.

The four key competitive priorities were identified as: cost, quality, delivery and volume flexibility [16]. Mass customisation should find the ideal trade-off between these conflicting priorities.

The studies conducted in the MC field cover sectors such as the food industry, electronics, large engineering products, mobile phones, personalised nutrition, homebuilding and foot orthoses [2]. The research done by Fogliatto et al. [2] revealed a significant increase in the academic investigation of the operational aspects of mass customisation; this finding was conducive to the relevancy of the scheduling method developed for the research.

Order penetration or Decoupling Point (DP) is an important aspect for mass customisation. The DP determines the degree of customisation, which influences the process planning for the product. The DP may occur at five different points in the supply chain: customer (mass production); retailer (minor customisation); assembler (partial MC); manufacturer (MC); and supplier (real-time MC) [17]. Alternatively, Squire et al. [16] separates the decoupling points into: full customisation (customer at design or fabrication stages); partial customisation (customer at assembly or delivery stages); and standard products (no customer involvement).

Decoupling at assembly and manufacturing stages can be accomplished via product platforms and modularisation [18]. A product platform can be defined as a group of shared components, modules, or parts from which a set of derived products can be developed and released [19]. Modularisation involves the decomposition of a product into subassemblies and components to facilitate the standardisation of components, while increasing product variety. Modularisation has proven to be a successful practicable facilitator for the implementation of mass customisation [2], [20].

Decoupling at the retailer level has been accomplished through postponement (Pp), of which there are two types: time postponement and form postponement. Time Pp is based on delaying the delivery of the completed product to the customer by initiating its manufacture only once the order is received (also known as make-to-order). Form Pp involves moving the differentiation downstream the supply chain such that the product is already in a semi-finished state and completion thereafter is initiated once the specific order is received [2], [21].

Fogliatto et al. [2] stated that the complexity of decoupling at the supplier stage has deferred its implementation in practice thus far.

The research undertaken aimed to facilitate the characteristics identified in this section. The product implemented for the research in Section 3.3 is to be customised by the customer before manufacturing is initiated, using inventory already present in the production system (as explained in Section 5.5). The DP pertinent to the research would be at the manufacturer stage, identified as ‘mass customisation’ by Tien et al. [17]; alternatively, the research would be placed under ‘full customisation’ as per the definition indicated by Squire et al. [16]. The concept of modularisation was employed for the fixture design implemented (Section 3.4). The DP implies that the manufacturing system must be capable of adapting to a high product variety, which necessitates the reconfigurable fixtures for which the fixture manufacturing cell was proposed. This places the FxMC within the mass customisation field, as was intended.

2.3 Jigs and Fixtures

The FxMC was primarily focused on the management of reconfigurable fixtures which can facilitate MC. The types of reconfigurable fixtures that exist were investigated. The fixture design research field

was reviewed to ascertain the state-of-the-art and to properly implement the most suitable reconfigurable fixture type in the research undertaken.

A fixture is a device used to physically locate, hold and support a workpiece during a manufacturing process; this may include machining, welding, assembly, inspection and testing. Jigs perform a similar task, while additionally guiding the cutting tool during machining operations [3]. Fixtures are an essential factor in the quality, productivity and cost of a manufacturing process. The design and fabrication of fixtures in a manufacturing system can make up 10 – 20 % of its total cost [4], while a large portion of rejected parts are said to be attributed to poor fixture design [22]. Fixture operations are generally outsourced to an off-site facility to improve the cost-effectiveness of the activity [5]; however, this method is not conducive to a mass customisation production system, due to the concurrent requirements of the fixtures and customised parts. Outsourcing fixture activities would affect the responsiveness of the manufacturing system due to the detachment of the respective activities; this is the issue that the on-demand FxMC aims to address.

Computer-Aided Fixture Design (CAFD) tools are used for the design of fixtures. CAFD is an amalgamation of Computer-Aided Design (CAD) and Computer-Aided Manufacturing (CAM) tools developed for fixture design purposes. Recent approaches to fixture design include Case-Based Reasoning (CBR), Genetic Algorithm (GA) and Neural Networks (NN) [23].

Dedicated fixtures are individually designed for use on a specific workpiece undergoing a particular manufacturing process [3]. Dedicated fixtures are used in mass production, where batch sizes are large enough to warrant the investment of time and capital for a specialised fixture. However, dedicated fixtures are not applicable to the different product types of mass customisation. Reconfigurable fixtures provide a solution to this problem; these include conformable and modular fixtures [24].

Conformable fixtures are designed to hold various types of irregularly shaped parts. These include pin-array fixture technology (Figure 2.1) and phase-change materials (Figure 2.2) [22].

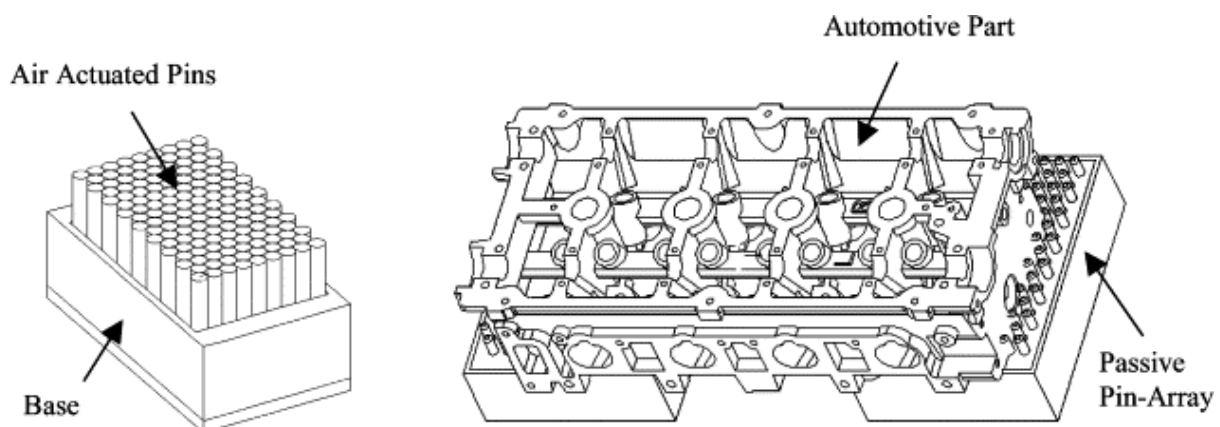


Figure 2.1: Pin-array type fixture [25]

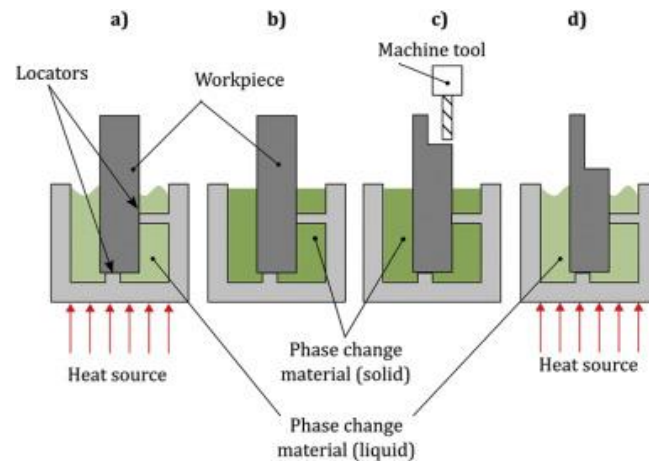


Figure 2.2: Phase-change material type fixture [26]

Modular fixtures are the most widely used reconfigurable fixture type [4]. Modular fixtures include grid hole, T-slot and dowel pin. Modular fixtures provide a limited number of combinations in comparison to other reconfigurable fixture types; and compromised stability, accuracy, and efficiency of the fixtures in comparison to dedicated fixtures. However, the performance in those respective categories is superior to the other fixture type (dedicated and conformable, respectively), such that a compromise in flexibility and operability is provided. An IMAO Corporation® grid hole modular fixture is shown in Figure 2.3.

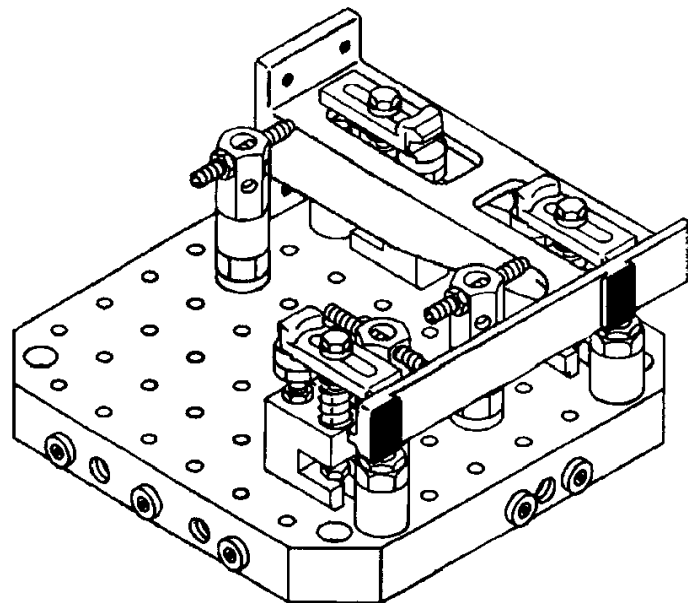


Figure 2.3: IMAO Corporation® grid hole modular fixture [3]

The modular fixture approach has been implemented for both FMS and mass customisation. Müller et al. [27] developed a system with robot manipulators in place of static modules for the handling of large aircraft parts, where lightweight modules for handling, joining and measuring were used together with an assembly platform; this represents an advancement of the modular fixture idea. A simpler modular fixture design is shown in Figure 2.4. Wallack and Canny [28] developed an adaptive fixture vice, with adaptability via translation along the X-axis. The fixture was capable of conforming to two-dimensional objects.

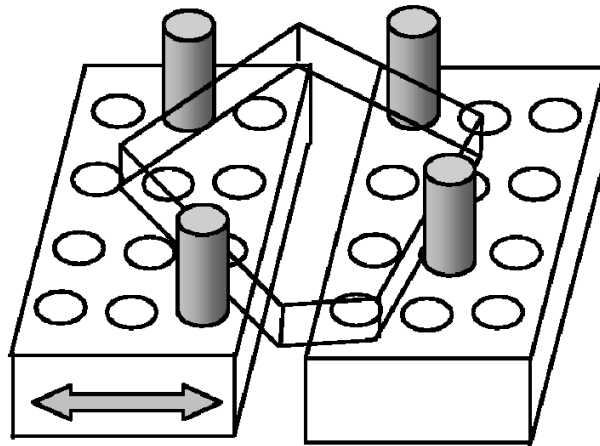


Figure 2.4: Adaptive fixture vice [28]

The fixture design employed in the research undertaken (Section 3.4) was analogous to those displayed in Figure 2.3 and Figure 2.4. A modular approach was chosen due to its cost-effectiveness, applicability to industry and suitability to the cellular manufacturing paradigm (as mentioned in Section 2.5). The modularisation technique was also pertinent to the mass customisation decoupling point (established in Section 2.2).

Bi et al. [12] noted that modular fixtures were identified as useful in mass customisation applications; however, the scheduling of how to utilise the modular fixture components efficiently in production planning had yet to be investigated. This finding provided evidence of the research gap in the field of study, and the importance of developing the research for the modular fixture type in particular. Further evidence of the gaps in research are elaborated through the scheduling studies reviewed in Section 2.8.2.

The fixture design process was investigated to provide a background for the fixture design implemented in the research (Section 3.4). There are generally four phases in the fixture design process: setup planning, fixture planning, unit design, and verification. These phases are summarised in Figure 2.5.

The main requirements in fixture design are summarised in Table 2.1.

The Physical requirement of fixtures to accommodate the workpiece geometry is in agreement with the necessity of the research to manage reconfigurable fixtures that can adapt to the changeable geometries of customised parts. The contribution of the scheduling method for the fixture manufacturing cell (Chapter 6) is in agreement with the Affordability requirement. The fixture reconfiguration time (or assembly/disassembly time for the modular components of the fixture design described in Section 3.4) is minimised through the techniques used in Stage I (Section 6.7) and Stage II (Section 6.8) of the scheduling method. The idle time for the manufacturing process associated with the part for that fixture is minimised through Stage III (Section 6.9) of the scheduling method.

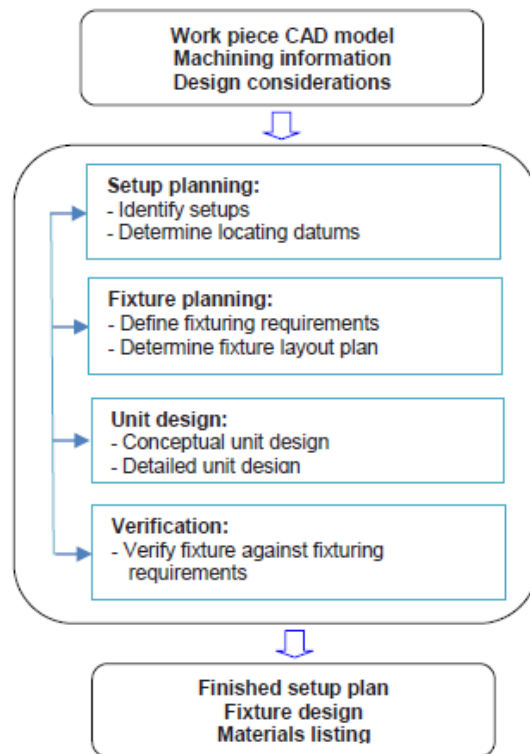


Figure 2.5: Steps to fixture design [29]

Table 2.1: Requirements of fixture design [30]

Requirements	Examples
Tolerance	– Locating tolerances of fixture should be in agreement with design tolerances for parts.
Physical	– Fixtures should be able to accommodate the workpiece geometry and weight. – Machining areas of the part should be accessible by the associated process.
Affordability	– Fixture cost should be kept to a minimum. – Fixture assembly/disassembly times should be kept to a minimum. – Fixture operation time should be kept to a minimum.
Constraint	– Fixture should ensure minimum moment and force equilibrium of workpiece. – Fixture stiffness should be sufficient to avoid deformation of either fixture or workpiece.
Usability	– Fixture weight kept to a minimum. – Fixture should not cause surface damage of workpiece at interface. – Fixture should provide tool guidance for machining of the workpiece. – Fixture should prevent erroneous workpiece setups. – Fixture should assist in guiding machined chips away from the current process.
Collision prevention	– Fixture should avoid tool path/fixture collisions. – The fixture should avoid workpiece/fixture collisions apart from the required interface. – The fixture should avoid fixture-fixture collisions, apart from the required interface.

2.4 Reconfigurable Manufacturing Systems

Mass customisation requires the advantages of both job shops (low volume, high variability) and Dedicated Manufacturing Lines (DML) (high volume, low variability). Literature on the subject predominantly focuses on FMS as a facilitator of merging these advantages [2]. However, the concept of Reconfigurable Manufacturing Systems (RMS) provides an alternative solution that aims to minimise the disadvantages of FMS. RMS encourages the customisable flexibility of the manufacturing system, as opposed to the general flexibility of each machine (as is the case for FMS) [31].

A RMS is defined as a manufacturing system that is designed with the intent of being able to rapidly change its structure (including hardware and software components) in order to quickly respond to market or regulatory changes, by being able to adjust its production capacity and functionality within a part family [31].

Flexibility is exhibited by both the FMS and RMS paradigms. Flexibility in manufacturing is described as the capacity of a system to change and adopt various positions or states in response to changing requirements; with minimal penalty in time, effort, cost and performance [32]. The degree of flexibility of FMS and RMS differ; FMS focuses on the flexibility of the individual machines to be able to produce a wide variety of parts, whereas RMS restricts itself to a flexibility range for a given part family. The high investment cost associated with FMS is circumvented by the RMS paradigm, since the degree of flexibility of the machines used is reduced; for instance, tool changes can be minimal and some dedicated machines can remain in the system [33].

Figure 2.6 is an idealised graph of the Manufacturing System Cost vs. Production Rate for FMS, RMS and DML. The graph shows a sharp increase in manufacturing cost of FMS for increased capacity, whereas manufacturing cost of RMS increases steadily toward a much higher capacity.

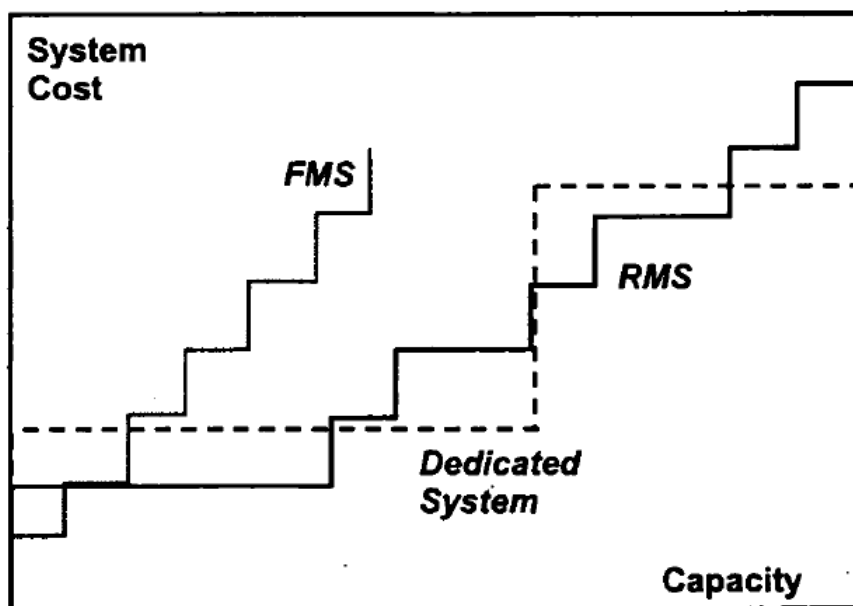


Figure 2.6: Idealised Cost vs. Capacity graph for FMS, RMS and DML [31]

Koren and Shpitalni [33] compiled a list of ten types of flexibility in manufacturing systems; these are as follows:

- 1) **Machine Flexibility:** Several operations executed without set-up adjustment.
- 2) **Material Handling Flexibility:** Numerous possible paths between all machines available.
- 3) **Operation Flexibility:** Various processing plans available for part manufacturing.
- 4) **Process Flexibility:** Part families that can be manufactured without significant set-up modifications, i.e. part-mix flexibility.
- 5) **Product Flexibility:** Convenience (with regard to time and cost) of adding new products into the current product mix.
- 6) **Routing Flexibility:** Number of feasible routes available to the various part families.
- 7) **Volume Flexibility:** Ability to increase/decrease production volume within the available capacity without reduction in profitability.
- 8) **Expansion Flexibility:** Convenience (with regard to effort and cost) of expanding capacity and/or capability through physical changes to the system when necessary.
- 9) **Control Program Flexibility:** Ability of a system to run autonomously through automated machines and system control software.
- 10) **Production Flexibility:** Numerous part families can be produced without major capital investment in new equipment.

The definitions exhibit overlap for the flexibility types. Expansion Flexibility is closely aligned to the flexibility proposed for RMS. This type of flexibility is intended to increase the lifespan of a manufacturing system, as adaptation over years (or decades) can be catered to.

A manufacturing system that intends to adopt the advantages of RMS should exhibit the six key characteristics of a reconfigurable manufacturing system, as identified by Koren and Ulsoy [34]; these are as follows:

- 1) **Customisation:** System or machine flexibility is limited to a single product family.
- 2) **Convertibility:** The ability to readily modify the functionality of prevailing machines and systems to match the new requirements.
- 3) **Scalability:** The ability to readily adjust production capacity by including/excluding resources or modifying the components of the system.
- 4) **Modularity:** The components are designed to be interchangeable and capable of being manipulated for alternative production schemes.
- 5) **Integrability:** The ability to quickly and precisely integrate the various modules into a single functioning system.
- 6) **Diagnosability:** The ability to easily detect and repair faults in the system to minimise defects.

The on-demand fixture manufacturing cell is capable of facilitating a RMS, as the concept is beneficial to the objectives of the manufacturing system type.

The FxMC can provide Process Flexibility and Product Flexibility through the flexibility of the fixtures implemented in it. Volume Flexibility, Expansion Flexibility and Production Flexibility can be facilitated through the reconfigurability of the fixtures used, in that the fixture capacity is capable of handling more part types and part volumes than the number of fixtures available.

The FxMC would enable a mass customisation production system to exhibit the six characteristics of the RMS. This claim is founded on the cell providing a central, separate hub for reconfigurable fixtures to be handled. The flexibility and changes to the rest of the manufacturing system would require only minor changes to the FxMC; these would primarily regard storage capacity increases and modifications to the fixture type used (if the new part family is significantly altered).

2.5 Group Technology and Cellular Manufacturing

The concept of the on-demand FxMC was based on separating the fixture activities in a specialised cell; this concept was derived from the manufacturing philosophies of Group Technology (GT) and Cellular Manufacturing (CM). GT was also pertinent to the grouping of similar parts to the same fixture in Stage I of the scheduling method (Section 6.7).

Group technology is a philosophy that exploits the similar characteristics of multiple problems to obtain a single solution that satisfies their shared requirements. In the manufacturing context, GT involves grouping similar parts into part families [35]. Group Technology is implemented in manufacturing systems through the concept of cellular manufacturing. CM involves aggregating similar machines or processes into cells, each of which is specialised for the production of a part family [36].

Group technology integrates the benefits of mass production in a batch production environment [37]. The aim of GT aligns with the objectives of mass customisation (Section 2.2). The implementation of RMS can be executed through a CM system. Therefore, a cellular manufacturing system can provide a suitable platform upon which a mass customisation production system can be developed and investigated.

Cellular manufacturing can yield improvements by separating and specialising tasks in the production system. The advantages of CM include: reduced throughput time; reduced work-in-process; improved product quality; reduced response time for customer orders; improved material-handling efficiency; increased manufacturing flexibility; reduced unit costs; simplified production planning and control; reduced setup times; reduced finished goods inventory; reduced production floor space; increased machine utilisation; and reduced machine idle time [36], [38].

Disadvantages of CM include: high capital investment required for changeover; lost production time due to changeover; and specialised training for operators [36].

Reconfigurable fixtures can be implemented in cellular manufacturing with a high degree of effectiveness. The specialisation of cells means that modular fixtures can be customised within the domain of the cell [36]; this minimises the main disadvantage of modular fixtures, where the customisability of the fixtures are limited by the characteristics of the prescribed modules (Section 2.3).

A cell in a CM system should consist of: machine/s; a part conveying system; and the cell controller [38]. A U-shaped layout is preferred for the sake of unidirectional flow [36]. A CM system also creates a unit workflow policy through its operation. Unit workflow is a facilitator of lean manufacturing [39]. These characteristics were integrated into the FxMC layout designed for the research (described in Section 3.5).

Cellular manufacturing systems can be implemented with a degree of flexibility to minimise the effect of strictly isolated cells. At least one cell should be capable of processing any of the parts, should downtime or overcapacity for a certain part family arise. Provisions should be made for highly specialised machines for which availability in the manufacturing facility is limited, such that parts from other cells are directed to the specialised machine/s via their process planning [38].

The efficiency of a CM system is affected by the assignment rule used to create the part families. The earliest method for family formation is Product Flow Analysis (PFA). The Rank-Order Clustering (ROC) algorithm was developed later, which was then extended to the direct clustering technique. Mathematical programming techniques were also developed to form part families, one of which is the p-median model [35]. The p-median model associates parts to medians (selected parts that anchor each family), based on a distance measure for similarity. Minkowski and Hamming distance are two distance measures commonly used. The algorithm minimises the weighted average distances from the medians [40].

The machine arrangement in the cells of a CM system is crucial for optimal operation to be achieved. The positioning of the machines on the factory floor influences factors such as material flow and operator walking distance. Disregard for such factors can affect the efficiency of the cell and diminish the benefits intended from the cell layout [41].

A mathematical model developed by Bazargan-Lari et al. [41] was studied under the topic of Machine Layout Problem (MLP). This model is governed by the following constraints:

- **Non-overlapping condition:** to prevent overlapping of resources in the model.
- **Shop floor boundaries:** to define the factory floor space.
- **Closeness relationships:** to establish a safe working area around machines for the operators and material flow.
- **Location restrictions/preferences:** to factor in restricted areas (such as aisles) and give preference to certain areas (such as the inclination to leave very large machinery unmoved to save on cost).
- **Machine orientations:** to face machines in preferred directions and maintain relationships between machines that should be orientated in relation to each other.
- **Travelling cost:** to factor in the rectilinear distance the material would have to travel and the associated cost.

The model obtains the co-ordinates and aspect ratio (hence, orientation) of each resource. This is done using a combination of goal programming and simulated annealing [41].

The mathematical model is generally applied to cells with at least five resources. The cell described for the research (Section 3.5) did not contain enough resources to justify the use of the MLP model. However, the constraints listed for the model were used as a guide for the objectives of the cell layout.

2.6 Production Planning

Production planning involves the design of a production process that allocates resources effectively, to ensure that the production demands of the system are met [42]. The scheduling of the on-demand FxMC is an important contribution of the research. The inputs to the scheduling method are based on the parameters fed to it from the other activities of the production system. The production planning architectures that exist in current manufacturing systems was investigated to formulate the FxMC and its scheduling method within a framework that would be relevant for a MC production system. These paradigms also include the systems and concepts currently investigated for the fourth industrial revolution [1].

Production planning systems can be classified as either push systems or pull systems. Hopp and Spearman [43] define a pull production system as one that explicitly limits the amount of Work in Process (WIP) that can be in the system, whereas a push production system is one that has no explicit limit on the amount of WIP that can be in the system.

2.6.1 Push Production Systems

Material Requirements Planning (MRP) is a classical production planning and inventory control system. MRP aims to maintain sufficient levels of inventory to ensure that the required materials are available when necessary. As a push production system, the focus of MRP is to push the production of items such that inventory levels are kept to a minimum [43], [44].

Manufacturing Resource Planning (MRP II) is an expansion of MRP; it factors in influences from other functional areas of an organisation, to create a feedback loop that aims to comprehensively control the system [45]. MRP II is defined as a computer-based system for planning, scheduling, and controlling the materials, resources, and supporting activities required to satisfy the production demands [42].

Alternative production planning systems that evolved from MRP include: Enterprise Resource Planning (ERP); Customer-Oriented Manufacturing Management System (COMMS); Customer-Oriented Management System (COMS); and Manufacturing Execution System (MES) [42]. These systems were not considered for the research, due to their increased focus on business-oriented functions that were beyond the scope of the objectives.

Production planning systems are driven by the Master Production Schedule (MPS). The MPS is a plan that specifies which items are to be produced and when they are due. The MPS is formulated from a combination of sales forecasts, customer orders, safety stock and internal orders. The Bill of Materials (BOM) is also a crucial element of MRP and MRP II. The BOM contains information on the materials and components required to create the products that must be produced. The BOM disassembles the products into their basic components to summarise the inventory requirements [44].

MRP and MRP II focus on push production. In a push production system, there is an emphasis on maintaining minimum inventory. The production is pushed out at a rate that is based on the processing ability of the first station of a production process. This first station processes the raw materials from inventory and pushes its completed work to the next station, regardless of whether that station is available or not. If any station other than the first is the bottleneck of the system (has the longest lead time) the WIP increases. This causes an increase in total lead time of the system [46]. A schematic representation of the push system is shown in Figure 2.7, demonstrating the flow from one workstation to the next, without intermediate control.

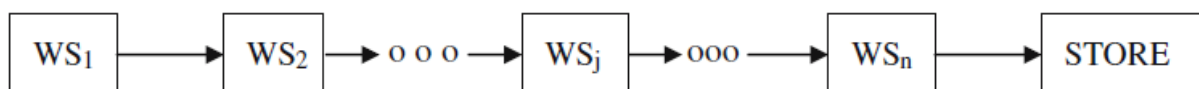


Figure 2.7: Push system schematic [46]

The Production Planning and Control (PPC) of a mass customisation production system with a fixture manufacturing cell is elaborated in Chapter 4, with reference made to MRP, MRP II and their

subsystems. The PPC system provides an overview of the functions at higher levels of the production system, from which the inputs to the scheduling method are retrieved. The higher levels of the PPC system were derived from push production systems, but the specific operation of the manufacturing system was derived from pull production systems (elaboration in Section 5.2).

2.6.2 Push Production Systems

Just-in-Time (JIT) manufacturing is an alternative system to MRP and MRP II. JIT is an example of a pull production system. JIT manufacturing systems have the primary goal of continuously decreasing (ideally eliminating) all forms of waste [47]. The method to achieve this goal depends on the ability of the system to deliver the right parts in the right quantity to the right place at the right time.

Kanban is a facilitator of JIT production systems. Kanban is the name given to the plastic cards used in the system, which contain the necessary information required for the fabrication and/or assembly of a product at each stage of its production process. The cards are attached to containers of a specific size, which can only hold the specified number of parts. Kanban is used in pull production systems [46].

There are two types of kanban systems:

- 1) Single card system, or Production Order Kanban (POK); and
- 2) Two-card system, where both a POK and a Withdrawal Kanban (WK) are used.

In the single card system, it is assumed that the distances between workstations are such that a single buffer can be used as both the outbound buffer of the previous workstation and the inbound buffer of the next workstation [46], [48]. A schematic diagram of the single card kanban system is shown in Figure 2.8 for a Workstation A and Workstation B.

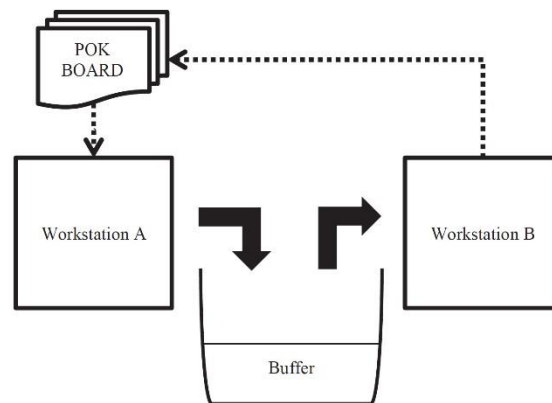


Figure 2.8: Single card kanban system [48]

The two-card system uses a separate outbound buffer and inbound buffer due to greater distances between workstations. The POK card is used for the same purpose as the single card system, while a WK card is added to retrieve containers from the outbound buffer. A schematic diagram of the two-card kanban system is shown in Figure 2.9 for a Workstation A and Workstation B.

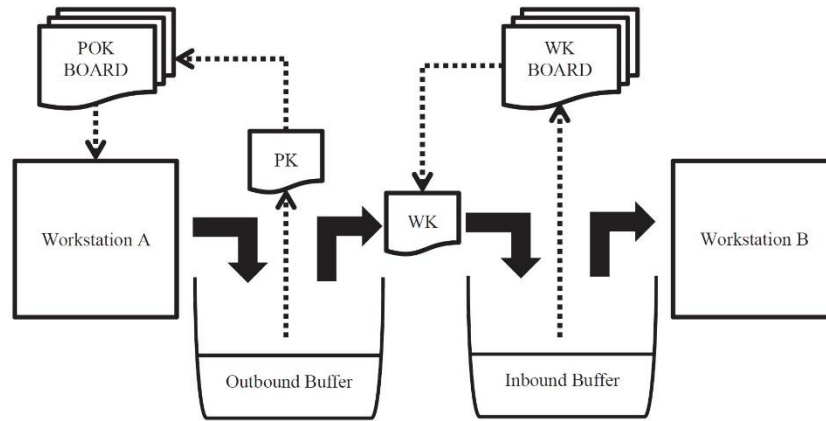


Figure 2.9: Two-card Kanban system [48]

The buffer storages for the workstations have limited capacity. As per the kanban system, a workstation is incapable of processing parts when either of its buffers are full; this is known as blocking. This is the mechanism by which kanban systems limit the WIP, and are thus classified as pull production systems. The limit on WIP (and hence the buffer capacity) is determined by the Toyota formula [49]:

$$K \geq \frac{DL(1 + \alpha)}{C} \quad (2.1)$$

Where:

K = number of kanbans,
 D = demand per unit time,
 L = lead time,
 α = safety factor, and
 C = container capacity.

The focus of the Toyota formula is to find the optimum number of kanbans for the system, as overstock occurs for a high value, and understock for a low value. Kekre and Karmarker [50] found that a decreased container size with increased kanbans lead to superior results. However, practical factors limit the ideal implementation in industry.

Lean production is defined as the production of goods or services that is accomplished with minimal buffering costs [43]. This means that there is minimal waste in the system as a whole, producing the required products by utilising the minimum of total resources. Buffering may refer to traditional waste, such as slow and unreliable machines; it may also include variability of the system, as this is a significant cause of waste in systems that rely on precise forecasts that vary in practice [43].

Lean production and JIT principles require a high standard in manufacturing. Decreased set-up times for reduced batch sizes, and stable and reliable production operations are prerequisites. Implementation and maintenance of these objectives can be difficult to achieve in practice. Human errors can be reduced through higher degrees of automation [51].

JIT, Kanban and lean manufacturing reveal the improvements in manufacturing efficiency obtainable by reducing buffering and batch sizes. The operation of the FxMC layout implemented in the research (Section 3.5) and the production system workflow (Section 3.6) are based on JIT and lean manufacturing. A kanban policy is implied through the unit workflow that was employed; fixtures and parts are retrieved one at a time. This is imposed through the instructed dispatch of parts and fixtures.

JIT characteristics are observed via the workflow, which is dependent on the completion of the prior operation; fixtures are not buffered, but idle time is minimised for improved utilisation of both fixture and workstations instead.

2.7 Optimisation Methods

A major contribution of the research undertaken was that of an optimisation model (or method) for the scheduling of the FxMC. A mathematical model is an abstract model that describes a real-world scenario in the language of mathematics [52]. Mathematical optimisation involves obtaining the best solution, with regards to the given criteria, from a set of alternatives [53]. Thus, an optimisation model is a mathematical model from which an optimal solution is to be found.

2.7.1 Formulation

The generalised form of an optimisation model can be stated as [54]:

Maximise/Minimise:

$$f(x) \tag{2.2}$$

Subject to:

$$g_i(x) \leq 0, \quad i = 1, \dots, I, \tag{2.3}$$

$$h_j(x) = 0, \quad j = 1, \dots, J, \tag{2.4}$$

Where:

$f(x)$ = objective function

$g_i(x)$ = inequality constraints

$h_j(x)$ = equality constraints

The objective function indicates the quantity to be optimised. Constraints limit the solution space by limiting the conditions for which the objective value can be evaluated. Decision variables are elements that are varied by the solution procedure in order to obtain the optimal objective value [53].

The objective function value may be either maximised (maximisation problem) or minimised (minimisation problem). The objective function can be either linear or non-linear. A linear objective function with all linear constraints is a Linear Programming (LP) problem. The LP problem becomes an Integer Programming (IP) problem if all decisions variables are integer-valued. The LP problem becomes a Mixed Integer Linear Programming (MILP) problem if some of the decision variables are integer-valued and some are continuous; this is the category of problem formulated in Section 6.9 for the research [53], after the initial Mixed Integer Programming (MIP) formulation (which contained non-linear constraints).

Feasible solutions are obtained when the decision variables are set to values that produce a valid solution, with the solution search space defined by the objective function and constraints [53]. The optimal solution is the best of the feasible solutions (as per the minimisation or maximisation criterion). The problem formulated in Section 6.9 is a minimisation problem (minimises total idle time).

2.7.2 Solution Techniques

The solution techniques for optimisation problems can be divided into: exact methods; heuristics; and metaheuristics [55].

Exact methods are used to obtain the optimal solution to an optimisation problem. These techniques are computationally expensive due to the analysis required to find the global optimum for a given problem. Therefore, exact methods are applied to Non-Deterministic Polynomial-Time (NP) complete problems and problems of minimal size. Exact methods include the Branch and Bound (B&B) method and Dynamic Programming (DP) [56].

Branch and bound is an enumeration technique. The method involves partitioning the solution search space into separate segments via branching, and then evaluating the solution generated at the partition against the bounded solution. For integer programming, the optimistic bound (or lower bound for a minimisation problem) is generally produced from the linear programming relaxation of the IP problem. The pessimistic bound (or upper bound for a minimisation problem) is generally produced from the first integer-feasible solution found, and successively updated for superior integer solutions thereafter. Branches are pruned (or fathomed) when either no solution is found at a node, or the solution is inferior to the pessimistic bound. The solution search at a branch is also fathomed when an integer-feasible solution is found at a node, thus implying that no superior solution is obtainable for that partition. The algorithm terminates the search at that partition when the branch is pruned, and backtracks to another node to continue. New branches are created when a favourable (superior to the current optimistic bound) non-integer solution is found, from which two branches are created to explore the integer values either side of that non-integer. Promising non-integer solutions also form the updated optimistic bounds. The algorithm is terminated when all nodes of the tree have produced solutions that are either: integer solutions; or a fractional solution inferior to any integer-feasible solutions found elsewhere. The optimal solution is the best integer-feasible solution obtained during the search [57].

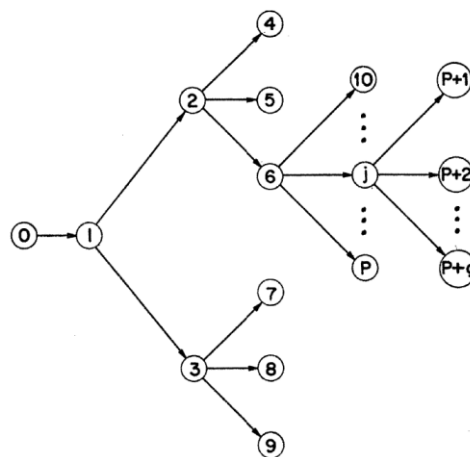


Figure 2.10: Branch and Bound tree representation [58]

Lagrangian relaxation is an alternative method of finding the pessimistic bound. Modifications to B&B include the branch-and-cut, and branch-and-price algorithms; these both combine the B&B algorithm described, with other techniques for fathoming nodes [57].

Dynamic programming is also an enumeration technique. The method divides the main problem into a series of sub-problems. The sub-problems are solved to optimality, then related to a larger sub-problem via a recursive relation. The sub-problems continue to expand in size until enough information is obtained from the sub-problem solutions to ascertain the optimal solution of the main problem [59].

Heuristics in the scheduling field include priority rules (or dispatching rules), and algorithms designed for specific problem types [60]. Heuristics place emphasis on minimal implementation time and computational expense, at the expense of solution quality. The solutions found are good, feasible solutions that are not guaranteed to be optimal, i.e. near-optimal [55]. The degree of closeness to the optimal value may vary significantly [61]. The dispatching rules implemented for scheduling problems include First-In First-Out (FIFO), Earliest Due Date (EDD) and Shortest Processing Time (SPT) [62]. Problem-specific heuristics include the bottleneck shifting method used for the Job Shop Scheduling Problem (JSSP) [63].

Metaheuristics include Simulated Annealing (SA), Tabu Search (TS), Genetic Algorithm (GA), Ant Colony Optimisation (ACO), and Particle Swarm Optimisation (PSO). Metaheuristics are advanced heuristics that search the solution space for the global optimum, while escaping from local minima [64].

Simulated annealing is derived from the physical annealing process for solid materials [65]. The temperature variable is initially set to a high value (simulating a high temperature) and is gradually decreased (cooled) at a specified rate. The high temperature corresponds to a high probability of acceptance for the solutions explored, before the cooling gradually decreases that probability until only good solutions fulfil the criterion [60].

Tabu search is a search method similar to SA. However, TS searches the solutions space by creating a tabu list of undesirable search directions, to avoid being restricted from exploring solutions outside the local optima. However, this does not guarantee that TS can always escape local minima [66].

Genetic algorithm has proven to be a prevalent metaheuristic in the scheduling field [67]. GA is an evolutionary algorithm that evaluates a population of solutions simultaneously [68]. A fitness function is formulated to assess the quality of the solutions in a population against the objective function value. Low quality solutions are deleted and the remaining solutions are used to produce new solutions for the next generation. The new solutions are created by either crossover (combining the attributes of two current solutions) or mutation (editing a current solution). Several rules for crossover and mutation may be chosen for implementation, which affect the effectiveness of the algorithm for the particular problem. The procedure continues until the termination criterion is reached [65]. GA is advantageous for vaguely structured problems, but the computation time required may be relatively high in comparison to problem-specific methods when these exist [66].

Ant colony optimisation mimics the behaviour of ants, by creating artificial ants that explore the solution space and leave pheromones that draw other ants to a promising solution [69].

Particle swarm optimisation is another algorithm inspired by nature. The initial solutions are generated at random and represented by particles moving at some velocity. New solutions are gained based on the movements of the particles, with consideration given to past experience of the particles. The method

has displayed quick convergence and simple implementation, but it is relatively new to scheduling problems [70].

Other metaheuristics are problem-specific, and so are not defined under a distinct category [65].

Heuristics and metaheuristics have generally outperformed exact methods for classical scheduling problems (such as shifting bottleneck heuristic and GA, respectively, for the JSSP) [71]. These methods produced optimal to near-optimal solutions in reduced time, and for larger problem sizes. The early formulations of classical scheduling problems were solved with exact methods (such as B&B for the aforementioned JSSP) [72]. The classical formulation was essential to understanding the problem structure and providing a benchmark optimal solution upon which the performance of subsequent methods could be compared [73]. This was a pertinent observation for the research undertaken.

The problem investigated in the research represented a novel undertaking. The structure of the manufacturing system (Section 3.6) was formulated specifically for the on-demand FxMC in relation to a MC production system. Thus, it was decided that an exact method approach would be appropriate for the initial formulation of the problem. The formulation led to a MILP model that was solved with a B&B algorithm (see Section 6.9). The research also provided a heuristic (Section 6.10) that was developed from the problem structure identified through the exact method formulation.

2.8 Scheduling

Scheduling is a decision-making process within the manufacturing industry that deals with the allocation of resources to tasks over specific time periods. The goal is to optimise one or more objectives, which involves the minimisation or maximisation of a measurable parameter/s [74].

Scheduling in a manufacturing system is associated with production planning (see Section 2.6). Figure 2.11 describes the information flow within a manufacturing system, from production planning to shop floor control, with the scheduling aspects integrated therein [74].

Scheduling utilises the information provided to it to manage the execution of tasks at the lower levels [74].

Scheduling methods are divided into two categories [74]:

- **Deterministic:** The job data consists of predetermined parameters, from which a definite schedule can be determined for the given job.
- **Stochastic:** The job data is unknown and subject to change, from which the schedule must be continuously updated until job completion.

Deterministic schedules can be solved to optimality with exact methods, whereas stochastic schedules generally require heuristics or metaheuristics to deal with the increased complexity of these problems [74].

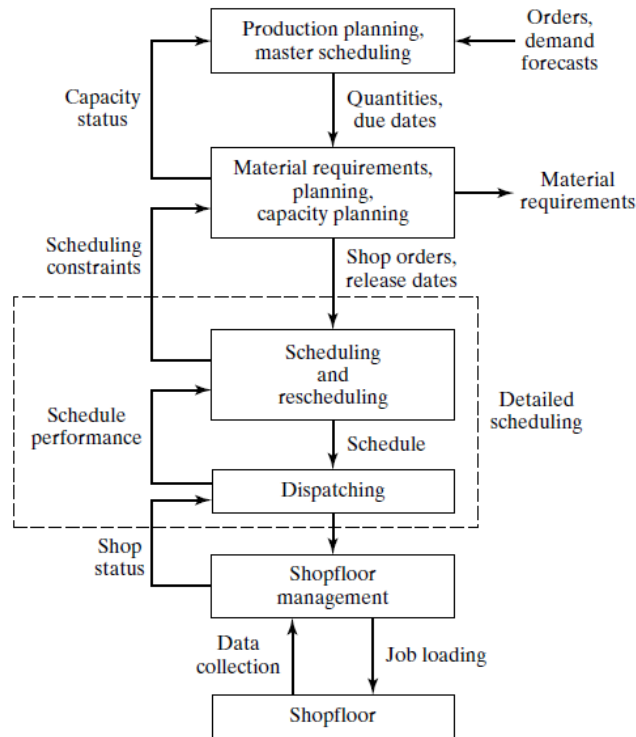


Figure 2.11: Information flow diagram in a manufacturing system [74]

Pinedo [74] lists the prevalent objective functions in scheduling problems as: makespan; maximum lateness; total weighted completion time; and total weighted tardiness.

The exact method approach decided upon (in Section 2.7.2) warranted a deterministic scheduling formulation. The predictability of the manufacturing system behaviour for a deterministic approach provided the opportunity to focus on developing the problem structure, instead of the real-time adjustments required for a stochastic approach.

2.8.1 Job Shop Scheduling Problem

The job shop scheduling problem is an important classical scheduling problem in the manufacturing field, as demonstrated by the studies conducted on the problem itself and modifications thereof since the 1950s [75].

The classical JSSP involves the scheduling of n jobs ($J = \{1, 2, \dots, n\}$) on m machines ($M = \{1, 2, \dots, m\}$). Each job is comprised of a sequence of N operations ($O = \{1, 2, \dots, N\}$); these can be considered machine-job mappings. An operation must be processed on its assigned machine for the duration of its processing time p . Pre-emption is prohibited, which means that an operation must be completed without interruption. The classical JSSP does not permit parallel processing, which means that operations are uniquely assigned to machines and time periods [75].

The disjunctive Mixed Integer Programming (MIP) model created by Ku and Beck [76] for the JSSP is presented below. The model was based on the formulation developed by Manne [77], which is one of the earliest models published for the JSSP.

$$\text{Min } C_{max} \quad (2.5)$$

Such that:

$$x_{ij} \geq 0, \quad \forall j \in J, i \in M \quad (2.6)$$

$$x_{\sigma_h^j, j} \geq x_{\sigma_{h-1}^j, j} + p_{\sigma_{h-1}^j, j}, \quad \forall j \in J, h = 2, \dots, m \quad (2.7)$$

$$x_{ij} \geq x_{ik} + p_{ik} - V \cdot z_{ijk}, \quad \forall j, k \in J, j < k, i \in M \quad (2.8)$$

$$x_{ik} \geq x_{ij} + p_{ij} - V \cdot (1 - z_{ijk}), \quad \forall j, k \in J, j < k, i \in M \quad (2.9)$$

$$C_{max} \geq x_{\sigma_m^j, j} + p_{\sigma_m^j, j}, \quad \forall j \in J \quad (2.10)$$

$$z_{ijk} \in \{0, 1\}, \forall j, k \in J, i \in M \quad (2.11)$$

Where:

The variable x_{ij} is the integer start time of job j on machine i ; the non-negative integer p_{ij} is the processing time of j on i ; the list $(\sigma_1^j, \dots, \sigma_h^j, \dots, \sigma_m^j)$ denotes the processing order of operations for job j on m machines; the binary variable z_{ijk} is equal to 1 if job j precedes job k on machine i . The Objective Function (2.5) minimises maximum completion time C_{max} , i.e. makespan. Constraint (2.6) ensures the start time is non-negative. Constraint (2.7) ensures the operation sequence is upheld. Constraints (2.8) and (2.9) ensure that there is only one job per machine for any given time period. V is a number large enough to ensure validity of (2.8) and (2.9). Constraint (2.10) ensures that the makespan is at least the largest completion time of the final operation of every job. Constraint (2.11) enforces the binary condition of z_{ijk} .

Solution methods for the JSSP include the B&B algorithm, disjunctive graph, priority rules and local search methods [75]. The shifting bottleneck heuristic is a prevalent heuristic solution method for the JSSP. This method creates a graph with conjunctive arcs only, analyses the machines to be scheduled, and determines the most disruptive disjunctive arc that could be implemented to continue constructing the graph, i.e. the bottleneck. The heuristic then optimises the bottleneck condition in isolation, such that it is no longer the bottleneck for the main problem [78].

The Flow Shop Scheduling Problem (FSSP) is a derivation of the JSSP, where jobs exhibit the same sequence as each other, i.e. every job utilises the same machines in the same sequence, thus relaxing the classical JSSP [79]. Johnson's rule is an algorithm developed to minimise makespan for the FSSP [75]. Johnson's rule schedules the jobs based on the duration of the operation time and the machine it relates to. For two machines, the algorithm continues selecting the shortest operation times from the list (updated after every iteration to eliminate the shortest time) and places them either at the start (if related to first machine) or the end (if related to second machine) of the schedule, until the schedule is complete.

The Flexible Job Shop Scheduling Problem (FJSSP) is another alternative to the JSSP, where multiple identical machines are available for processing of jobs, such that machine-job mappings are not necessarily exclusive [79]. Jobs can be processed in parallel, and decisions are made for which machine

is utilised for a given operation. The complexity of the FJSSP is increased in comparison to the JSSP, which has led to numerous recent studies that focus on this variation of the problem [73].

The problem formulated for the research (Section 6.3) was advised by the JSSP. The production system workflow (Section 3.6) exhibited characteristics of a flow shop, whereby a consistent job flow from Cell 1 to Cell 2 was observed. However, the final problem structure was unique to the research problem, and the formulation specific to it. Nevertheless, the classical scheduling problems did provide an initial background, which advised the final scheduling method that was developed.

2.8.2 Research Studies

Research studies pertaining to scheduling and optimisation of relevant production systems were investigated. A selection of the studies reviewed are presented below.

Evolutionary algorithms have emerged as the state-of-the-art in the scheduling field. Scheduling studies within the manufacturing environment commonly comprise of the JSSP and modifications thereof [73]. Algorithms explored include: ACO, co-evolutionary algorithm, classifier system, differential evolution, estimation of distribution algorithms, evolutionary programming, evolution strategies, evolvable hardware, GA, genetic programming, interactive evolutionary computation, linkage learning GA, memetic algorithm, parallel GA, probabilistic model building GA, and PSO [63].

Birgin et al. [80] created an extension to the FJSSP by generating the order of operations for the jobs with an arbitrary acyclic graph instead of using a linear order. The objective function was to minimise the makespan. A list scheduling algorithm was used together with a beam search method to find the optimal solution.

Mencia et al. [81] developed a genetic algorithm for JSSP, with weak Lamarckian evolution used to enhance chromosomes, together with search space narrowing to improve efficiency. The objective was to minimise makespan.

Ku and Beck [76] investigated the performance of MIP models for the classical JSSP, as it was found that evaluation of these models with modern software packages had not been adequately investigated. The authors revealed that MIP models for scheduling problems are prevalent in current literature, despite the onset of advanced metaheuristics. Two disjunctive models, a time-indexed model and a rank-based model were tested and compared; along with the use of multi-threading and parameter tuning to improve performance. The software packages utilised for the tests were CPLEX®, GUROBI® and SCIP®. The results varied, depending on the problem size and the software used. It was concluded that MIP models can be solved for moderately-sized problems (up to the 15-job×15-machine problem) in reasonable time (within 3600 seconds) with modern scheduling software. The 20×20 problem was tested but not solved within the time limit for the instances tested.

Jalilvand-Nejad and Fattahi [82] used a MILP model to solve a FJSSP with cyclic jobs. The manufacturing system implemented a kanban policy. A total cost function was created, which included setup cost, delay cost, finished product holding cost and WIP holding cost. The objective was to minimise total cost. A genetic algorithm and simulated annealing algorithm were developed for larger-sized problems, due to the NP-hardness of the JSSP. The GA outperformed the SA algorithm.

Scheduling within the cellular manufacturing field was investigated.

Sakhaii et al. [83] created an integrated MILP model for a dynamic CM system with unreliable machines, together with a production planning problem. The objective was to minimise the costs of machine breakdown and relocation, operator training and hiring, inter-intra cell part trip, and shortage of inventory.

Liu et al. [84] used a Discrete Bacteria Foraging Algorithm (DBFA) to simultaneously solve cell formation and task scheduling in a CM system. The objective function was to minimise material handling costs, and both fixed and operating costs of machines and workers. The DBFA was compared to a GA, and the results of the former were found to be superior in this study.

Raminfar et al. [85] developed an integrated model for production planning and cell formation. A MIP model was developed to solve the production planning and cell formation problems for a CM system, simultaneously. The objective function was to minimise inter-cell material handling cost, machine operating cost, production set-up costs and part inventory cost.

Studies regarding the utilisation of fixtures in an optimisation model were of particular interest to the research.

Thörnblad et al. [7] conducted a study on a multi-task cell, defined as a FJSSP. A time-indexed formulation of the problem was used. Side constraints factored in preventative maintenance, fixture availability and unmanned night shifts. A generic iterative procedure was used to solve the problem, together with a non-generic squeezing procedure. The objective function was to minimise the total weighted tardiness, where the weighting increased as the delay for a job increased. The fixture constraints in the study were to assign a particular fixture to a particular job, and to limit the number of fixtures of each type.

Yu et al. [9] conducted a study on a RMS with multiple process plans and limited pallets/fixtures. The goal was to determine the input sequencing and scheduling of the RMS. A deterministic schedule was created to be tested on, which included multiple process plans for the various jobs. The problem was solved using a priority rule based scheduling approach. Multiple objectives were investigated: minimising makespan, minimising mean flow time, and minimising mean tardiness. The practical constraint of releasing a job only when the relevant pallet/fixture was available, was included in the study.

Doh et al. [10] expanded on the work of Yu et al. [9] by conducting a study on a FJSSP with reconfigurable manufacturing cell. The decisions were based on: finding operation/machine pairs for processing parts; sequencing of parts to be sent through the reconfigurable manufacturing cell; and sequencing the parts assigned to each machine. The objectives and solution technique were the same as for the precursor work, which yielded sub-optimal solutions. The study also similarly regarded fixtures by constraining the part flow based on availability of fixtures for parts.

Da Silva et al. [86] conducted a case study on the scheduling of assembly fixtures in the aeronautical industry. The fixtures comprised of multiple workstations to hold large aircraft parts during the assembly process. The arrangement of workstations resulted in adjacency constraints, which prevented

access to available workstations on the factory floor. Mathematical models were developed to optimise the production scheduling for the assembly problem, yielding improved results over the traditional methods in practice.

Wong et al. [8] solved a resource-constrained assembly JSSP with lot streaming technique by utilising a GA with job-based order crossover. The model objective was the minimisation of total lateness cost. Resource constraints were used to place limits on the tools and fixtures used in the process. The resources were recyclable in the system.

Metaheuristics (evolutionary algorithms in particular) are prevalent in recent scheduling research studies. However, the relevance of exact methods and integer programming formulations remain current. Integer programming methods and exact solution techniques are required to find the optimal solutions and benchmarks for novel problem formulations. The JSSP and its alternatives are the scheduling problems that are predominantly investigated. The JSSP provided a background from which the MILP model formulation (Section 6.9) was advised; however, the intricacies introduced through reconfigurable fixtures and the manufacturing system workflow meant that the scheduling method was designed as a problem-specific solution. The details of the scheduling method formulation is elaborated in Chapter 6.

Optimisation research in the reconfigurable fixture area predominantly comprised of improvements in the design of the fixtures themselves [22], [23]. Scheduling studies that did consider fixtures were scarce; the studies found were included in this section. It was established that fixture utilisation in a manufacturing environment was limited to regarding fixtures (standard, not reconfigurable) as a constant resource through a single constraint. The research undertaken aimed to investigate this problem more comprehensively than the studies conducted to-date. The fixture manufacturing cell required an optimisation method that considered reconfigurable fixtures, and the recirculation of those fixtures for customised parts; a problem of this type was not explored by the studies reviewed. Bi et al. [12] declared the need for efficient scheduling of modular fixture components with the manufacturing system; this finding was confirmed by the absence of equivalent studies corresponding to this problem in the literature.

2.9 Additive Manufacturing

Additive Manufacturing (AM) can facilitate the realisation of mass customisation in practice [87]. The technology could benefit the fabrication of custom modules for a reconfigurable modular fixture, such as the fixture implemented in the research (Section 3.4).

The American Society for Testing and Materials (ASTM) defines additive manufacturing as the “process of joining materials to make objects from 3D model data, usually layer upon layer” [88]. Additive manufacturing began as a form of rapid prototyping technology, but has evolved to become feasible enough for rapid manufacturing (in limited circumstances thus far).

Additive manufacturing exhibits advantages that are conducive to the realisation of mass customisation. Material wastage is minimal. Resource requirements are minimal; traditional resources (such as jigs, fixtures and cutting tools) not required. In the context of customisation, there are no geometric constraints in AM. Part design is not constrained by Design-for-Manufacture (DFM) principles; and

separated parts can be made as one, instead. Products can be produced in batches of one, with minimal consequences other than the actual part requirements (marginal economies-of-scale influence) [89].

Disadvantages of AM are related to current technological limitations. The materials used (predominantly polymers) do not permit the production of large parts due to the lack of material strength. The imperfections of the parts created exhibit rough surface finishes, due to the resolution and accuracy limits of current machines. The cost of AM is currently high (for machines and material), but commercial machines are becoming more affordable as the technology advances [89].

Additive manufacturing has the potential to simplify the traditional supply chain [90]. Products can be designed with fewer components; this reduces warehousing, transportation and packaging. The modularity of the machines (one machine can produce the entire part) means that factory size can be reduced; thus, factories can be located closer to the customers, which further reduces transportation and property cost. The benefits of lean manufacturing and JIT principles can be exploited, due to the minimal inventory and material stock required [91]. The responsiveness of an agile supply chain can also be improved, which aligns itself with RMS characteristics (Section 2.4).

2.10 Summary

This chapter presented a review of topics relevant to the implementation of an on-demand fixture manufacturing cell. The mass customisation field was researched, together with manufacturing systems that could facilitate the idea (namely RMS and CM systems). It was found that the concept of the cell was appropriate for implementation in these manufacturing systems for mass customisation. The modular fixture was justified as an applicable design approach for implementation in the FxMC. Production planning systems were studied, with focus on the methods of applying push and pull systems, with advantages and disadvantages elaborated thereof. Optimisation methods were investigated for preparation of the scheduling method that was developed (Chapter 6). The JSSP was presented as a basis upon which scheduling optimisation models are commonly constructed. Current research studies were reviewed; the relevancy of integer programming models and exact methods was confirmed, despite emphasis on metaheuristics for more established problems (such as the classical JSSP and its derivatives); the management and scheduling of reconfigurable fixtures had not been adequately investigated, from which the necessity of the research was justified. Additive manufacturing was briefly discussed as a facilitator for mass customisation.

3 The Fixture Manufacturing Cell Concept

3.1 Introduction

This chapter provides an introduction to the concept of the fixture manufacturing cell and the purpose it intends to serve in a mass customisation production system. The test product, fixture design, cell layout and production system workflow for the research implementation are described thereafter.

3.2 The Concept

The concept of a fixture manufacturing cell was based on the requirement of customised parts to be served with reconfigurable fixtures on-demand. Fixtures are required to adapt and respond to customer-demands as rapidly as the customised products. The current industrial practice involves outsourcing jig and fixture production, due the activity being perceived as one of low added value [5]; this approach does not enable the responsiveness required for mass customisation. The FxMC provides a production resource located in the manufacturing facility where the management of reconfigurable fixtures can be handled in-tandem with the Product Manufacturing System (PMS). The benefits of RMS for mass customisation was discussed in Section 2.4, with cellular manufacturing noted as a facilitator of RMS in Section 2.5. The FxMC utilises the advantages of cellular manufacturing to provide the production system with the flexibility and efficiency of a specialised cell for reconfigurable fixtures.

The FxMC concept provides a solution to the on-demand requirements of a mass customisation production system. However, the concept necessitates an appropriate decision support system for the management and scheduling of the cell, such that the PMS can be adequately assisted by it. The general scheduling decisions of a fixture manufacturing cell can be summarised by the flowchart displayed in Figure 3.1.

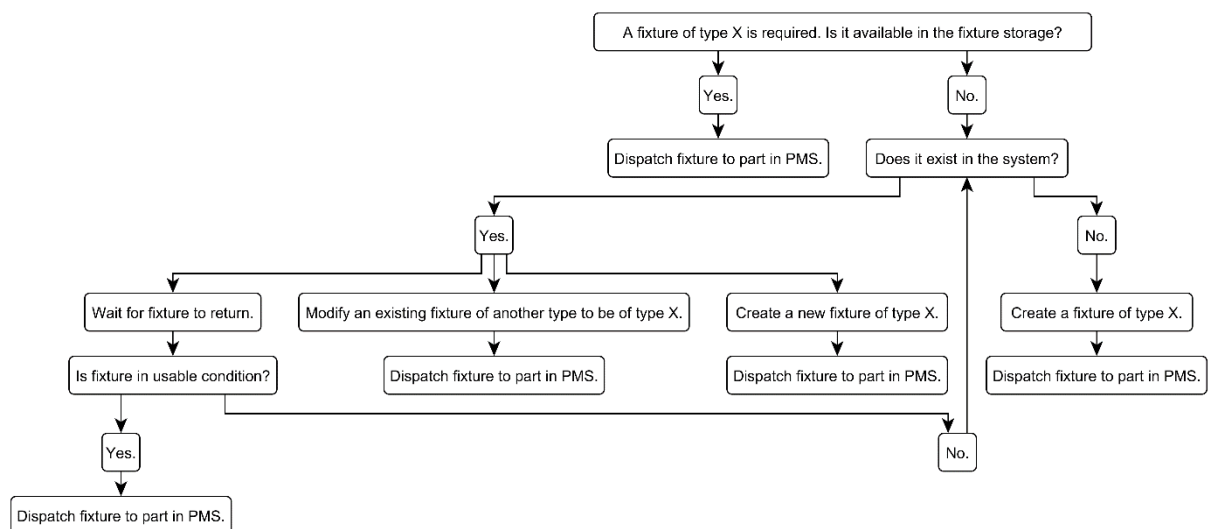


Figure 3.1: A general FxMC decision flowchart

The management of the FxMC affects the efficiency of both the PMS and the cell itself. A high fixture inventory would ensure reliable fixture availability for the PMS, but lead to high resource expenditure

and holding costs for the FxMC. A low fixture inventory would ensure leaner fixture management, but jeopardise reliability of production in case of high product demand. The research focused on the management of fixture reconfigurations in particular, where a predetermined fixture inventory is dispatched and modified in an optimal sequence, such that parts in the PMS can be produced with minimal time wastage.

3.3 Test Product

In order to examine fixture management in a mass customisation system, it was decided that a test product should be selected, from which the reconfigurable fixture, cell layout and production system could be structured and investigated.

The test product was decided to be an engraved plaque. This provided a basic, two-dimensional part undergoing an automated manufacturing process, which would require a simple fixture design to secure it. The product would exhibit customisability of both: border shape; and engraving design.

Customisability of border shape was the characteristic that necessitated reconfigurable fixtures. Variations in the border shape required different fixture configurations to secure the part during the engraving process. Customisability of the engraving design added variation in the processing times of different products, such that fixture reconfiguration times and part processing times could be varied further.

The conceptual implementation of this test product necessitated Operation Flexibility, Process Flexibility, and Product Flexibility (see Section 2.4) from the manufacturing system.

3.4 Fixture Design

The fixture design used for the research is described in this section. The focus of the research was on the management of these fixtures in the production system, and not on the fixture design itself. The fixture design provided a conceptual platform upon which the research could be implemented and tested.

Modularity has proven to be a successful method of practical implementation for mass customisation; as suggested by Gershenson et al. [20] and Fogliatto et al. [2] with respect to parts; and Bi and Zhang [4] with respect to fixtures. This is true for cellular manufacturing systems, where reconfigurable modular fixtures can be designed for flexibility within a part family, producing fixtures with greater effectiveness for the particular task [36]. Thus, a modular fixture design was chosen for use in the research.

A grid hole base plate with dowel pin modules was the selected design; similar to those displayed in Figure 2.3 and Figure 2.4. The base plate comprised of a square block of dimensions 200 mm×200 mm×16 mm; with an 8×8 array of through holes, each 10 mm in diameter. Dowel pins of diameter 10 mm and length 40 mm were used as modules. Pin configurations were assembled upon the base plate by inserting the dowel pins into the base plate array in an arrangement that was appropriate

for securing the part. Figure 3.2 shows the pin configuration required to secure a large, triangular part. Figure 3.3, thereafter, shows how this part would fit within the pin configuration.

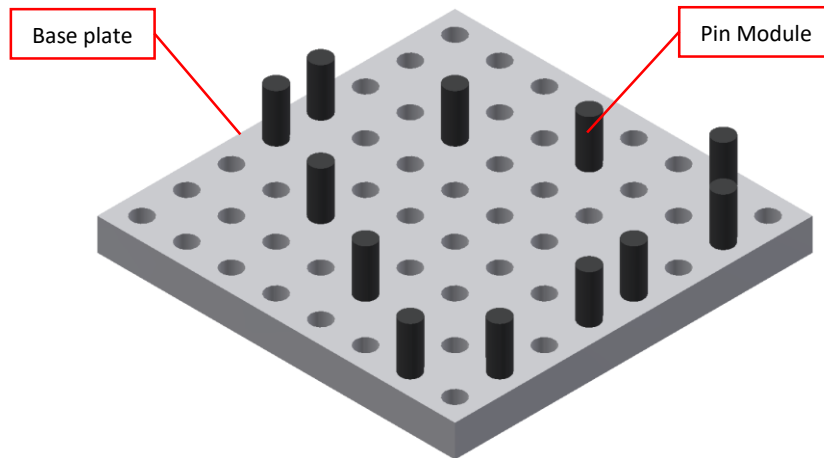


Figure 3.2: Pin configuration for large triangle part (isometric view)

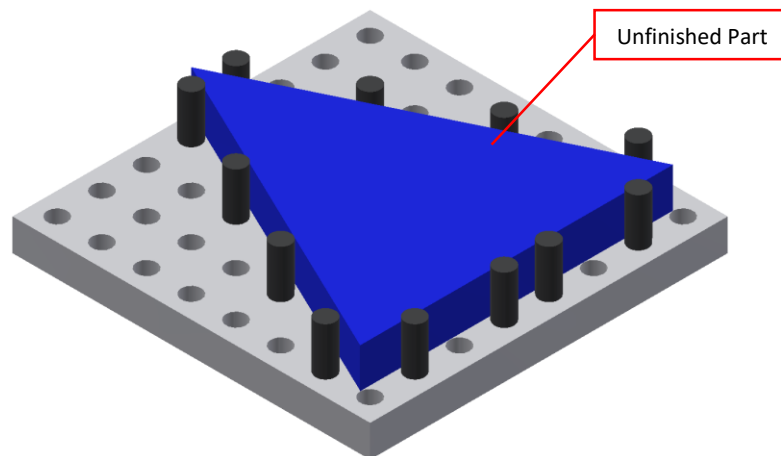


Figure 3.3: Large triangle part on fixture (isometric view)

The force applied by the tool in an end milling process can be calculated by Equation 3.1 [92]:

$$T = 0.05K_dF_fF_TBW + 0.007K_dD^2JW \quad (3.1)$$

Where:

T = thrust (N)

K_d = work material factor, based on material hardness (BHN)

F_f = feed factor, based on feed (mm/rev)

F_T = thrust factor, based on drill diameter (mm)

B = chisel edge factor for thrust, based on chisel edge to drill diameter ratio

W = tool wear factor, based on operation type

D = drill diameter (mm)

J = chisel edge factor thrust other, based on chisel edge to drill diameter ratio

A sample calculation was performed to investigate the force caused by an end milling process with 2.4 mm diameter drill bit, operated at a feed rate of 0.05 mm/rev on aluminium alloy material. The calculation is shown in Equation 3.2, with parameters obtained from tables in [92]:

$$T = 0.05(7000)(0.091)(2.02)(1.235)(1.1) + 0.007(7000)(2.4)^2(0.01)(1.1) \quad (3.2)$$

$$T = 90.51 \text{ N}$$

Equation 3.1 shows the thrust to be proportional to the diameter of the drill bit. A drill bit size of 2.4 mm for the engraver yielded a force of 90.51 N, which was deemed acceptable for the fixture design upon examination. The decreased drill bit size of engravers would also yield reduced vibrations in comparison to larger drill bits more commonly used for end milling processes (> 10 mm) [93]. The pin configuration, thus, prevented rotation and translation of the part in the X-Y plane. The design was deemed sufficient for the engraving process that the plaque would undergo.

The fixture design was limited by the resolution of the base plate array. The 8×8 dimensions of the array was arbitrarily selected, estimated from past designs [3], [4], [11]. The use of custom modules could have improved the resolution limitation of the base plate array. It was decided that pin modules would be used instead, for the purpose of simplifying the quantification of pin configuration comparisons for the scheduling method (Section 6.7). Despite the practical limitations of the fixture design, the base plate array could theoretically accommodate a total of 2^{64} unique pin configurations. Several of these pin configurations would be unusable in practice (such as those with one pin, or those with 64 pins).

The research considered pin configurations within the range of 8 – 16 pins per configuration. Only four pins were required to prevent rotation and translation in the X-Y plane for most shapes. The range of 8 – 16 pins was selected by testing observations of various arbitrarily-shaped parts upon a template of the base plate array; it was concluded that no more than 16 pins would be required to properly secure almost any part shape from translation or rotation in the X-Y plane. The exception to this conclusion was the circle shape, where prevention of rotation was independent of the number of pin modules used for its configuration. This special case required an interference fit with the dowel pin modules to prevent rotation via friction with the dowel pin module circumference.

The number of available pin configurations were calculated from the summation of the associated binomial coefficients, as shown in Equation (3.3).

$$C = \sum_{k=a}^b \frac{n!}{k!(n-k)!} \quad (3.3)$$

Where:

C = total number of unique pin configurations

n = total number of holes

k = number of pins evaluated

a = minimum number of pins used

b = maximum number of pins used

The total number of usable unique pin configurations available was calculated to be 7.1325×10^{14} — a very high number. While full customisability would theoretically mean an infinite number of possible configurations, the customisability of the fixture design was deemed sufficiently high for the purposes required for the research implementation.

The fixture design fulfilled the Physical and Affordability requirements (see Section 2.3). The Tolerance and Constraint requirements were fulfilled for the pin configurations that satisfied the

practical customisability afforded by the base plate array. The other fixture design requirements (Usability and Collision Prevention) are beneficial to the robust operation of the fixture, which was beyond the scope of the research.

Table 3.1: Fixture specifications

Parameter	Value
Fixture type description	Modular grid hole base plate design with dowel pin modules
Base plate dimensions	200 m×200 mm×16 mm square plate
Base plate array dimensions	8×8 holes (64 holes total); 10 mm diameter through holes
Dowel pin module dimensions	40 mm height×10 m diameter
Theoretical customisability of fixture	2 ⁶⁴ unique pin configurations
Pin range selected for research implementation	8 – 16 pins per configuration
Practical customisability of fixture	7.1325×10 ¹⁴ unique pin configurations

3.5 FxMC Layout

The research required the implementation of a proof-of-concept of the FxMC. This section describes the layout, equipment and general functions of the fixture manufacturing cell. The FxMC should be capable of manufacturing, storing, and modifying reconfigurable fixtures, before dispatching them to the PMS, after which they are returned to the FxMC for recycling in the system. Therefore, the FxMC was expected to comprise of:

- **Fixture Storage:** to receive, store and dispatch the fixtures.
- **Fixture Raw Material Inventory:** to store the raw materials required to create new fixtures.
- **Fixture Fabrication Station:** where new fixtures are created.
- **Fixture Reconfiguration Station:** where fixtures are reconfigured for the relevant part operation.
- **Control Station:** from which the operator is expected to control and monitor the actions of the cell.
- **Transportation System:** to transfer fixtures to their required locations in the production system, and recirculate them thereafter.

The cell layout was conceptualised using the resources available for the research. The FxMC layout schematic is shown in Figure 3.4.

The components and functions of the cell are explained with reference to Figure 3.4. The FxMC implemented in this research is a semi-automated, single-operator cell; as it requires a manual operator to perform tasks using mostly automated equipment. The cell layout is arranged to employ unidirectional workflow, which is preferred in manufacturing systems to improve efficiency and control [39]. The cell layout is structured such that workstations are within close range of the operator for any given task, allowing for efficient movement within the cell.

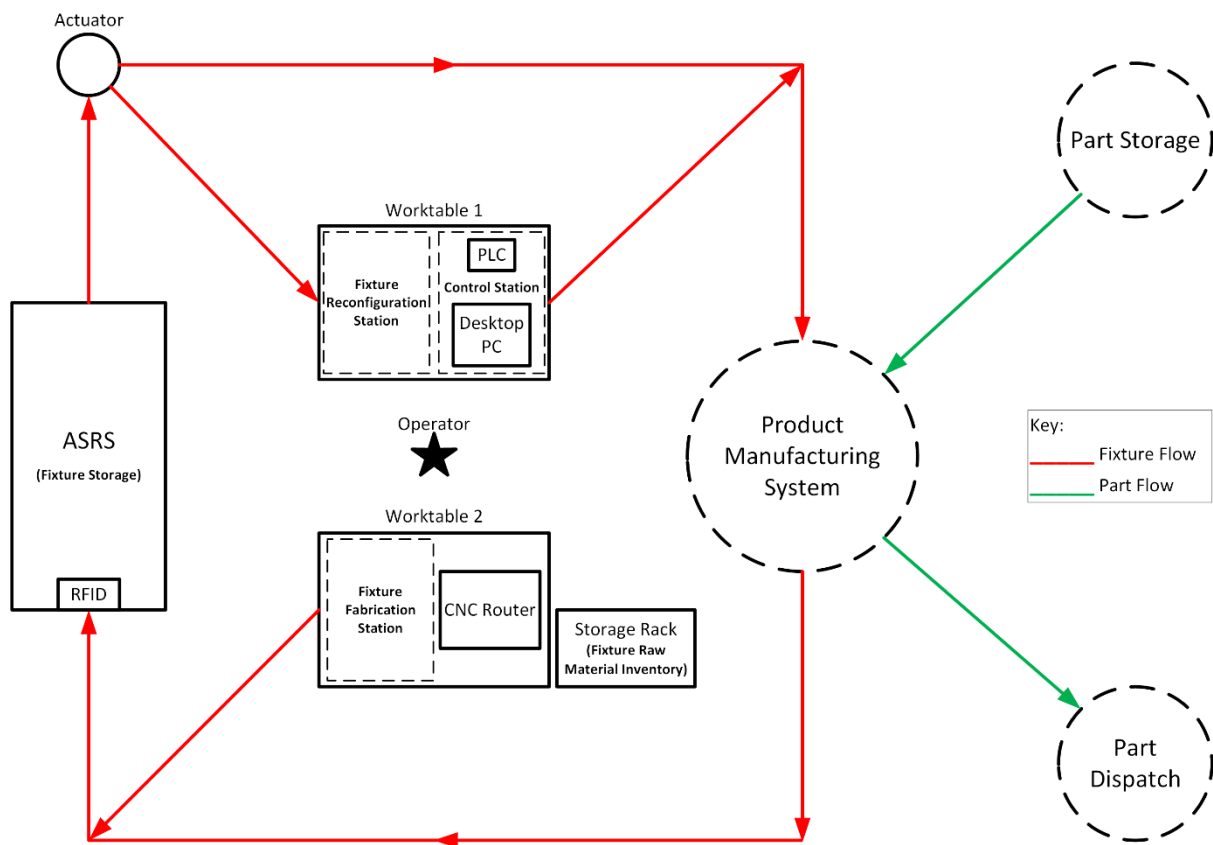


Figure 3.4: FxMC layout schematic

The cell utilises components and resources available for the research in the Mechatronics and Robotics Laboratory at the University of KwaZulu-Natal (UKZN). The workflow is performed by a group of conveyors, which represent the automated Transportation System of the cell. The fixture flow is shown in red, and the part flow is shown in green. The fixtures are transported on pallets.

An Automated Storage and Retrieval System (ASRS), with in-built Radio Frequency Identification (RFID) scanner, is used as the Fixture Storage element in the cell. The machine is used to store incoming fixtures and retrieve outgoing fixtures. The fixture pallets contain embedded RFID tags for identification via the RFID scanner at the input of the ASRS. This ensures that individual fixtures can be tracked during their circulation in the production system.

The fixtures are retrieved through the exit of the ASRS via its output conveyor, towards the actuator. The actuator directs the fixture along one of two paths:

- A direct path to the PMS if the fixture is already correctly configured for the succeeding part operation; or
- A path along the conveyor branch to the Fixture Reconfiguration Station if the fixture is to be reconfigured for the succeeding part operation.

The former is predominantly the circumstance of batch orders, which is expected to be a rare occurrence for a mass customisation system due to the unique customer demands that are anticipated. Thus, it was expected that most fixtures would arrive at the Fixture Reconfiguration Station at Worktable 1 due to

diverse product orders. The operator reconfigures the fixture to the required configuration by adding, removing or rearranging modules on the fixture base plate at this worktable.

The Control Station is also situated at Worktable 1, which is represented by the Desktop PC and Programmable Logic Controller (PLC) unit. The operator is able to control and supervise the ASRS, conveyors and actuator via the Control Station components. The Desktop PC provides the job schedule (with regards to fixture reconfigurations required) and the status of the fixtures in the system (either in storage or in circulation), amongst other information, via the Graphical User Interface (GUI).

The Fixture Fabrication Station is represented by Worktable 2. The worktables are located such that the operator can efficiently move between them. New fixtures are created at Worktable 2. A Storage Rack is used for Fixture Raw Material Inventory, which is where the raw materials for fabricating new fixtures are stored and retrieved by the operator. In the case of the fixture design implemented in the research, the raw materials would be solid square plates and dowel pin modules. The base plates would be manufactured by the CNC Router, which can drill the base plate array holes via the pre-programmed G-code uploaded to it; see Appendix C.1 for Standard Operating Procedure (SOP). The fixture is then assembled by inserting the required pin modules for its first part operation (see Appendix C.2 for SOP). This fixture would then travel directly to the PMS via the ASRS (for the purpose of initial identification via the RFID scanner), since no further modification would be required for that run.

The correctly configured fixtures travel to the PMS. The fixture is used to secure the part during the manufacturing process for the customised product to be made (see Appendix C.3 for SOP). The parts are retrieved from Part Storage, which contains unfinished parts or raw materials. The parts are processed in the PMS in a way that depends on the process planning for that customised product. The completed products are then dispatched to the Part Dispatch area for customer delivery, while the fixture returns to the ASRS to be recirculated in the system when required.

The FxMC layout was designed with consideration of the resources available for the research. The layout described formed the basis upon which the practical implementation was conducted. Improvements in efficiency could be made by substituting the manually operated tasks with fully automated components. However, single-operator manned cells do have notable advantages; particular with regard to cost savings, considering the initial capital investment required to convert a traditional manufacturing system into one that utilises GT principles [36]. Despite scope for increased automation, the structure and function of the FxMC layout remain valid for industrial implementation. Therefore, the FxMC layout described in this section was deemed satisfactory for the implementation of the fixture manufacturing cell in the research. The layout was used to structure the practical implementation of the FxMC in Chapter 4.

3.6 Production System Representation

The fixture manufacturing cell should be investigated in context of the production system within which it exists and the manufacturing process that it serves. This section presents the production system representation considered for the research. Figure 3.5 describes the workflow and subsystems of this production system.

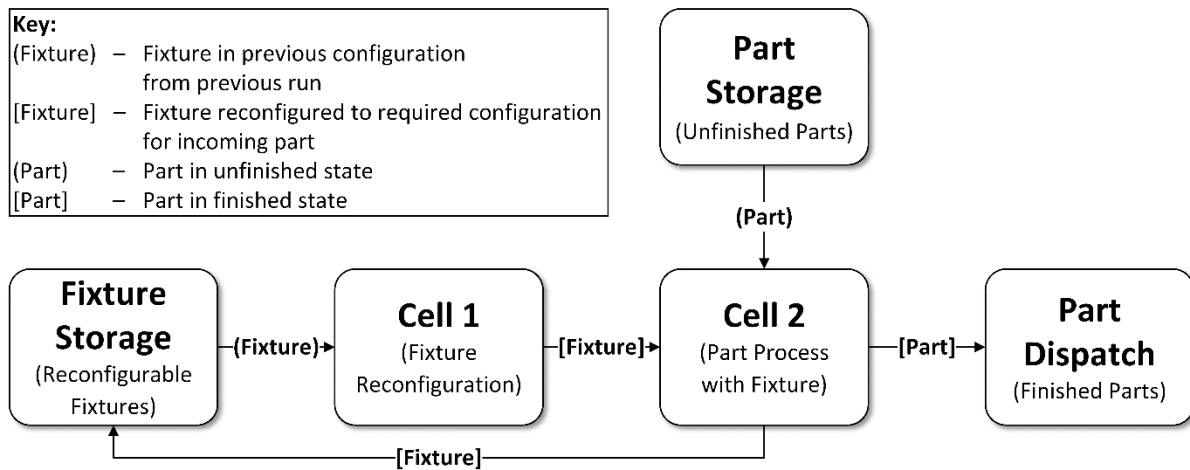


Figure 3.5: Workflow of production system representation

The production system described in Figure 3.5 depicts the representation of a mass customisation system implemented in the research. It was established in Section 2.4 that the FxMC enables a mass customisation system to exhibit the flexibility and listed characteristics of RMS. These features of RMS are aligned with the adaptability and responsiveness necessary for mass customisation to be achieved [94]. The RMS production layout can be structured as a CM system. CM is also conducive to mass customisation requirements and the implementation of reconfigurable fixtures (Section 2.5). Therefore, the CM paradigm was used to represent a mass customisation production system in the research.

The cellular manufacturing paradigm was employed for both the fixture manufacturing cell (Cell 1) and the product manufacturing system, which was condensed into a single part processing cell (Cell 2). ‘Part processing cell’ is thus used interchangeably with ‘product manufacturing system’ hereof. Furthermore, Cell 1 and Cell 2 are used interchangeably with ‘fixture manufacturing cell’ and ‘part processing cell’, respectively, hereof.

The specific structure and activities of the part processing cell can vary in scale and function, depending on the manufacturing system requirements. The part processing cell was implemented as only a representation of a mass customisation product manufacturing system in the research, where a single manufacturing process occurred.

The initial function of the fixture manufacturing cell is to fabricate new fixtures. However, the research was primarily concerned with how these fixtures are managed in the system thereafter. Therefore, the production workflow considered fixture reconfigurations as the main task of the fixture manufacturing cell.

The workflow in Figure 3.5 is described. The fixtures are sent to Cell 1 from Fixture Storage in the configuration from the previous use of that fixture. The fixture is reconfigured to the required configuration and then sent to Cell 2. The part process that requires the fixture to secure the unfinished part occurs in Cell 2. The unfinished part is sent to Cell 2 from Part Storage, and the incoming fixture is used to secure it for the operation. The part is then processed on the current fixture, while the fixture to be used for the next unfinished part is concurrently being reconfigured in Cell 1. The finished part is

then sent to Part Dispatch, while the fixture is sent back to Fixture Storage to be made available for recirculation.

The system relied on a synchronous relationship to exist between the operation times of the fixture manufacturing cell and the part processing cell, i.e. the fixture reconfiguration time (ρ_{ij}) and part processing time (τ_{ij}), respectively. Cell 1 reconfigures the fixture to be used for the next unfinished part while Cell 2 is processing the current part on the previously reconfigured fixture. Bottlenecking results from either cell being occupied with its operation while the other cell has completed its operation. This would result in either the reconfigured fixture being unable to advance to the part processing cell, or the incoming unfinished part waiting for its fixture to arrive from the fixture manufacturing cell. Buffering of the reconfigured fixtures after Cell 1 would circumvent bottlenecking. However, buffering would hinder the recyclability of the fixtures in the system. The reconfigurable characteristic of the fixtures mean that optimal performance is obtained when there are more fixtures available from which the most suitable can be selected. Buffering reconfigured fixtures before they are used would limit the availability of these fixtures for subsequent unfinished parts to utilise. This would necessitate an increase in fixture inventory, which would lead to high resource expenditure and low utilisation of the fixtures. These consequences would impede the purpose and advantages of using reconfigurable fixtures. Therefore, a just-in-time workflow policy was evident in the production system, which favoured the advantages of lean manufacturing principles, as discussed in Section 2.6.2. A fixture was only unavailable for selection in Cell 1 when being used in Cell 2. This exploits the recyclability and reconfigurability of the fixture, which are fundamental properties of the FxMC concept.

3.7 Summary

This chapter introduced the concept of the fixture manufacturing cell as a solution to providing reconfigurable fixtures to a mass customisation production system on-demand. The FxMC decisions were presented, with the scope of the research stipulated to focus on the reconfiguration of fixtures. This stipulation limited the FxMC scheduling method described in Chapter 6. However, Figure 3.1 does present a framework for a more comprehensive (perhaps stochastic) model to be constructed for the FxMC in future.

The test product was presented as a two-dimensional engraved plaque, which provided customisability of border shape and engraving design, from which fixture reconfiguration times and part processing times could vary.

The fixture design was based on the selected test product. The fixture was of the modular type, with a base plate and dowel pin modules for variable configurability. The fixture design was limited by the base plate array resolution, and the use of dowel pins instead of custom modules. Thus, the fixture was restricted to a finite number of pin configurations only, in order to satisfy tolerance and constraint requirements. However, it was established that the fixture was highly customisable, providing a platform on which an order of 10^{14} configurations could conceivably be constructed, depending on the size and shape of the ordered part. The fixture was also shown to be adequate for its physical requirements due to the low thrust forces exerted by engraving (< 100 N).

The FxMC layout was described. Despite the focus on fixture reconfigurations, the FxMC was designed to be capable of achieving the tasks presented by the decisions in Figure 3.1, such as fixture fabrication and fixture tracking. The semi-automation of the cell was a limitation on efficiency, but the structure and function presented by the layout and components remain valid. A manned cell also provides the advantages of cost-effectiveness and simplicity, which could improve robustness and ease of troubleshooting in practice. The FxMC layout placed emphasis on efficient workflow and operator convenience. The layout was used for the laboratory FxMC described in Chapter 4.

The production system workflow was presented. The RMS paradigm was employed through a CM production layout. A two-cell just-in-time workflow resulted from the production system, which was used as the research representation for a mass customisation production system. The production system promoted lean manufacturing and due to the absence of a buffer before the part processing cell and JIT unit workflow. This increased fixture utilisation and decreased fixture storage inventory, but increased susceptibility to delays caused by bottlenecking. The scheduling method in Chapter 6 was designed to minimise such delays.

4 Practical Implementation

4.1 Introduction

The fixture manufacturing cell required a proof-of-concept to be implemented, as stated by the objectives in Section 1.4. The corresponding objectives (1 – 3) necessitated the implementation of mechanical, electrical and electronic hardware, together with programmable software, to assemble and automate the cell. This chapter details the practical implementation of the cell (based on the FxMC layout in Section 3.4), with reference to the individual components that were employed.

A proof-of-concept of the fixture manufacturing cell was assembled in the Mechatronics and Robotics Laboratory at UKZN. The laboratory FxMC (or Lab FxMC) is shown in Figure 4.1.

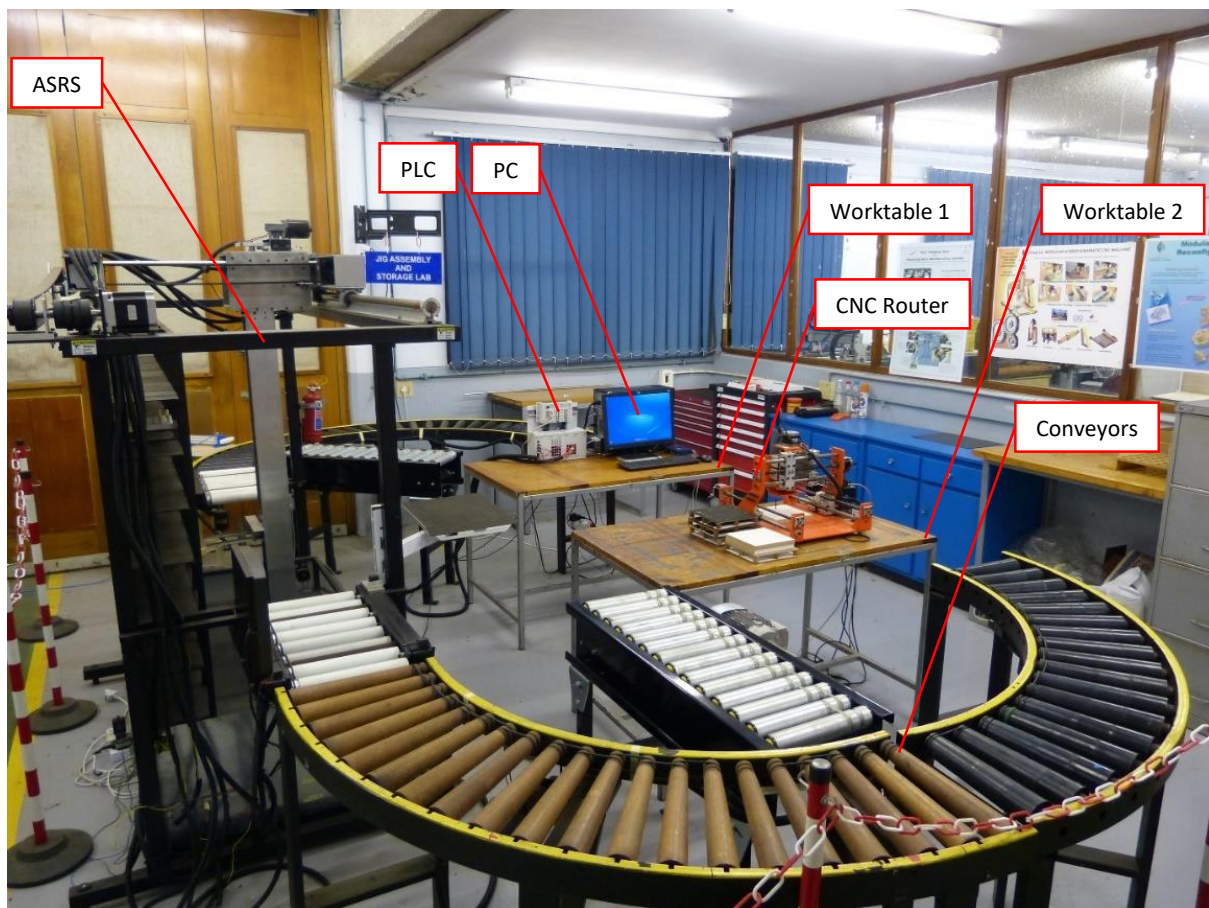


Figure 4.1: Lab FxMC

4.2 Existing Infrastructure and Modifications

The ASRS was an existing machine. The design was that of a single-aisle, single-rack layout. The storage rack comprised of 42 shelves: arranged in six rows and seven columns. A twin-fork gripper was used to pick and place pallets (which transported the fixtures). The motion axes provide linear motion in the X (along the aisle), Y (into and out of shelves) and Z (up and down) directions, with additional rotation about the Z-axis (to face the input and output conveyors). Motion in the X and Y directions were achieved through toothed belt and pulley drives; Z-axis linear motion was achieved with a ball-

screw and nut system, while its rotation was achieved through a Geneva wheel mechanism. The input and output conveyors were both driven by a band drive system. Stepper motors were used to drive the linear axes. DC motors were used to drive the Z-axis rotation and conveyors. Stepper drivers controlled the step and direction of the stepper motors. Transistor circuits were used to control the DC motors. The input conveyor was activated and deactivated by light sensors; a laser light with Light Dependant Resistor (LDR) was used. Limit switches were used to provide datum points from which the gripper position could be homed. The entire system was automated through an Arduino Due® microcontroller. Major refurbishments and amendments to the control system were made for functional operation to be achieved. The ASRS is shown in Figure 4.2.

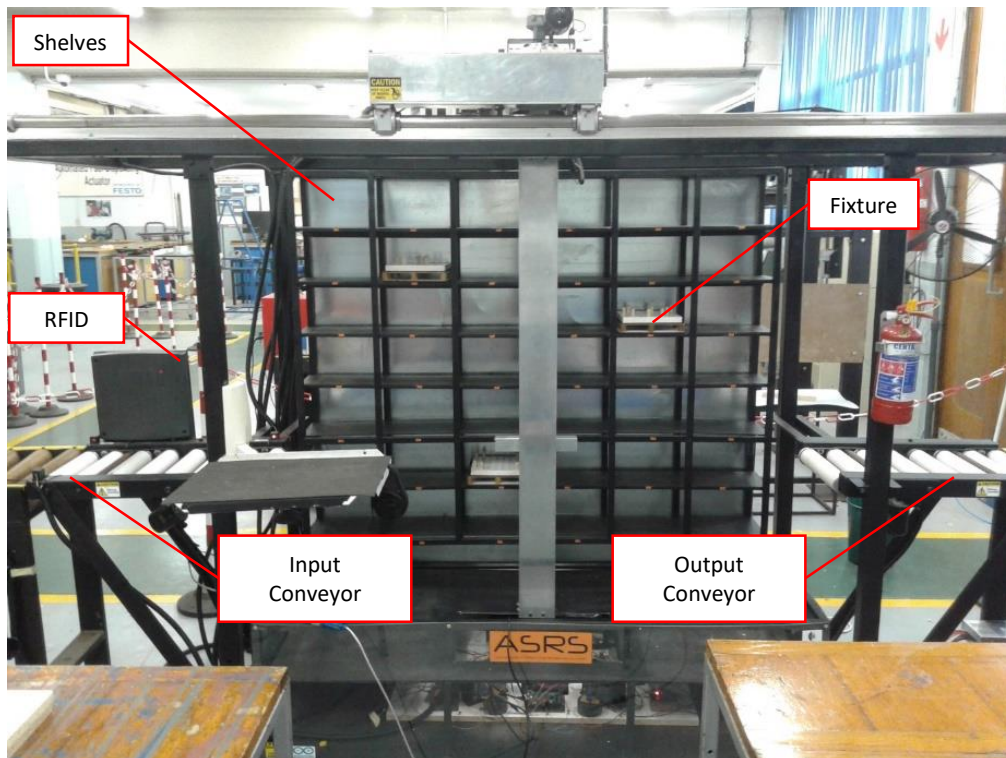


Figure 4.2: Automated Storage and Retrieval System used

The RFID scanner was mounted alongside the input conveyor to scan incoming pallets. RFID tags were embedded into the pallets. The tags were scanned upon entry to the ASRS, and the identification was exported to an Excel® spreadsheet via MATLAB®. No amendments were required to this function.

The ASRS was operated from a GUI designed in MATLAB®. The GUI provided the user with options to store and retrieve pallets, while updating the database accordingly. No amendments were required to this function. The ASRS GUI is shown in Figure 4.3.

The CNC router was an existing machine. The machine was a 3-axis device, with linear motion in the X, Y and Z directions. The motion axes were driven by a thread bar drive system, which substituted for a leadscrew drive system (with compromises in accuracy and speed). The X-axis was driven by two thread bar drives, which provided improved stability. The thread bars were each coupled to NEMA 23 stepper motors, which were controlled by Big Easy® stepper drivers and a CNC breakout board. The machine was connected to a PC via a 25-pin parallel port connection.

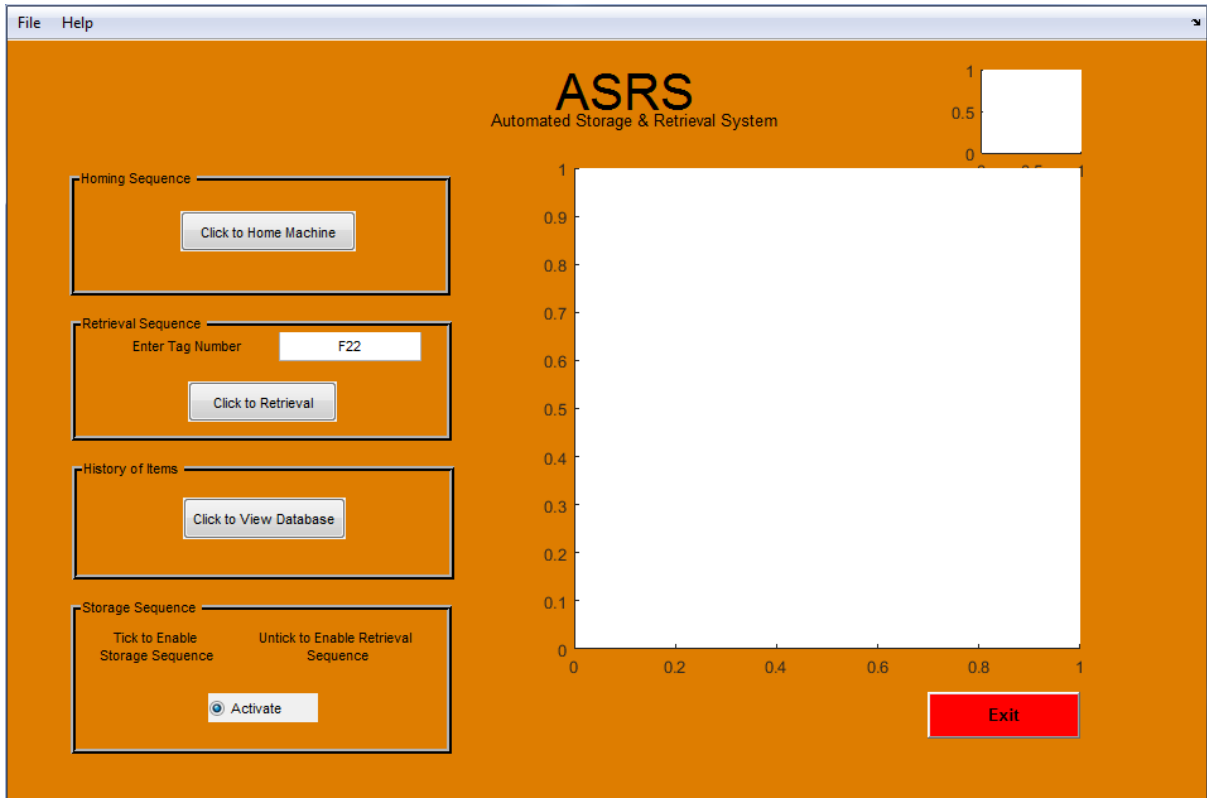


Figure 4.3: ASRS Graphical User Interface

Newfangled Solutions® Mach3® software was used to programme the machine. Minor refurbishments and amendments to the control system were made for functional operation to be achieved. The CNC router is shown in Figure 4.4.

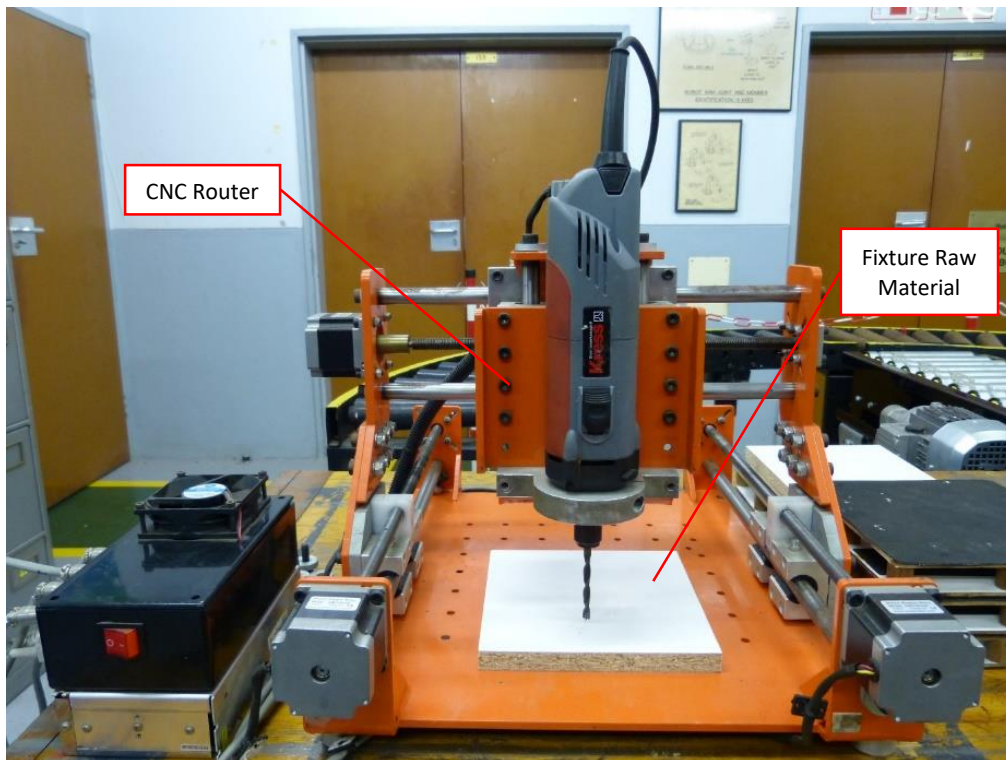


Figure 4.4: CNC router for fixture fabrication

The conveyor segments were existing structures in the laboratory. The conveyors comprised of five separate segments, denoted by the workflow function the conveyor facilitated (as denoted in Figure 4.5):

- From Fixture Fabrication (straight);
- To ASRS (curved);
- To Fixture Reconfiguration (straight);
- To PMS (curved);
- From PMS (curved);

The drive system for the curved conveyors were each of the band drive type, with DC motors responsible for every 1 m of conveyor length. The motors were initially not connected; the details of how this problem was addressed is discussed in Section 4.3. The DC motors were each coupled to the main drive bands by a pulley each. The pulleys had to be adjusted on their mounting plates for adequate band tension.

The two straight conveyors were purchased. A band drive system was used to drive the rollers via a three-phase motor for each segment. The motors were connected to a three-phase main switch in the laboratory.

The conveyor segments were arranged in a layout that best represented the FxMC layout described in Figure 3.4. A gap is shown on the right of the worktables in Figure 4.1, which is where the product manufacturing system would exist. Despite the curved conveyors and space constraints, the unidirectional flow of the FxMC layout was maintained for the Lab FxMC.

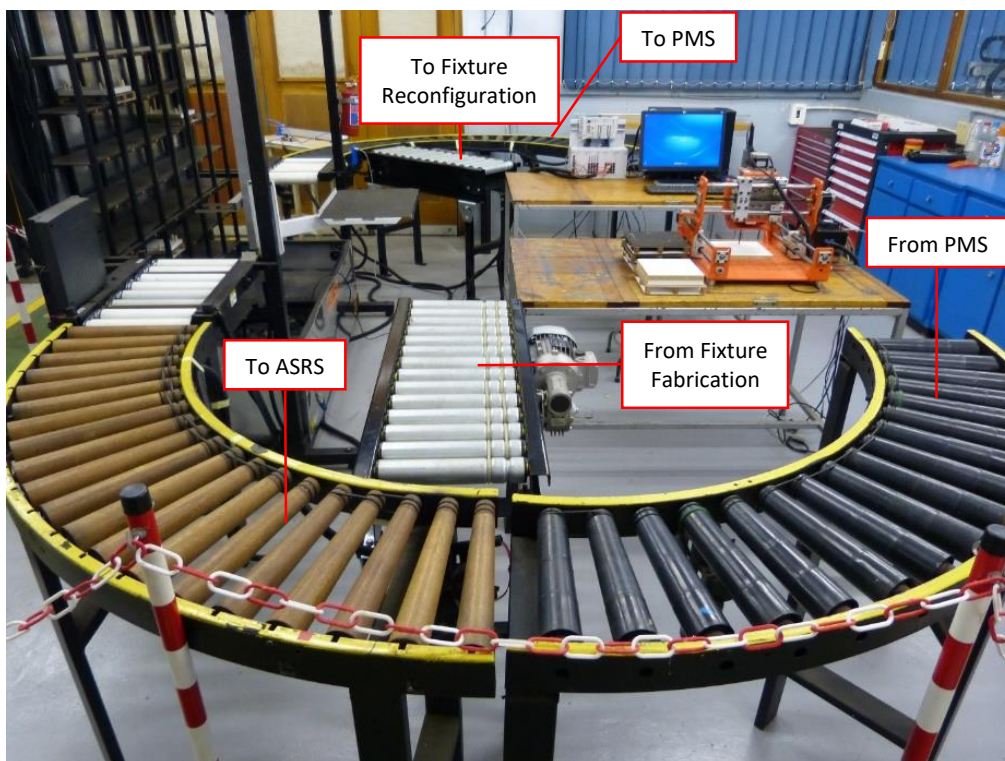


Figure 4.5: Conveyors

An example of the pallets used for the Lab FxMC is shown in Figure 4.6. The pallets were existing items; constructed from wood, with RFID tag embedded underneath the rubber covering. The size of the pallets advised the size of the fixture design in Section 3.4. The size of the pallets were constrained by the size of the shelves in the ASRS storage rack. The pallets were used to transport the fixture throughout the cell.



Figure 4.6: Pallet with embedded RFID tag

4.3 Conveyor System Electronics

The automation of the workflow in the cell necessitated the integration of electrical and electronic components to drive the conveyors. The curved conveyors were driven by 12 V DC motors via a band drive system. The motors were initially disconnected and unused. A 12 V DC Power Supply Unit (PSU) (see Appendix D.1) was used to transform the AC main voltage to 12 V DC voltage for the operation of the motors. The current rating of the PSU was 29 A. Testing revealed that the six motors could output a current of around 4 A each. Thus, the PSU current rating was deemed sufficient for the purpose.

The conveyors were automated with a Festo® Programmable Logic Controller (PLC) unit, which contained a 24 V DC output module. The relays were 24 V DC relays, which corresponded to the PLC output voltage rating. Thus, the relay coils were energised based on the 24 V DC output signal, which completed the circuit for the motors to the 12 V DC PSU.

Single Pole-Double Throw (SPDT) relays were used, as these were readily available and served the required purpose. The relays energised at 24 V DC and had a current limit of 10 A (see Appendix D.2), which was within the operational conditions of the motors. The single output (the motor) was connected to the Normally Open (NO) terminal. The Normally Closed (NC) terminal was not connected to a device; it was used only to turn the motor off. The 12 V DC PSU was connected to the Common (COM) terminal. The coils closed the internal circuit from the device to the motor (COM to NO) when energised by the corresponding PLC output port. The relay terminal connections are shown in Figure 4.7 for a single relay.

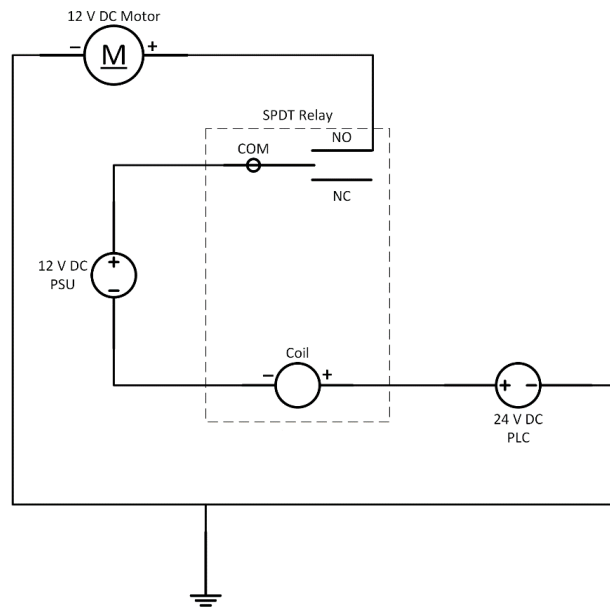


Figure 4.7: Relay terminal connections

The relay terminal connection in Figure 4.7 was duplicated for each relay, with the power supply and ground connections each linked in parallel. The relays and Printed Circuit Board (PCB) terminals were soldered onto a single segment of stripboard to facilitate the parallel connections. Figure 4.8 shows the top view of the circuit.

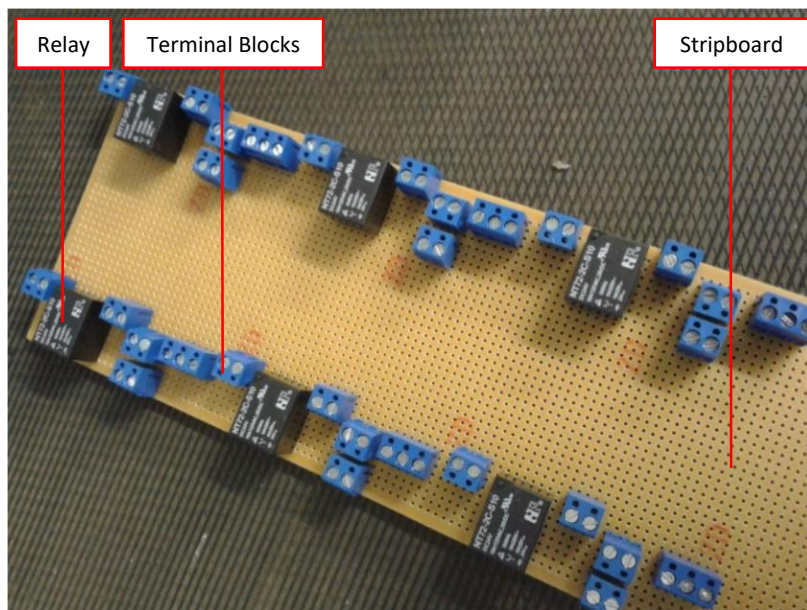


Figure 4.8: Soldered relay circuit

The stripboard was comprised of tracks arranged in columns, which provided a single electrical pathway along each track. The tracks had to be broken along the longitudinal axis of the relay to prevent a short-circuit occurring between the COM and NC terminals; this is shown in Figure 4.9.

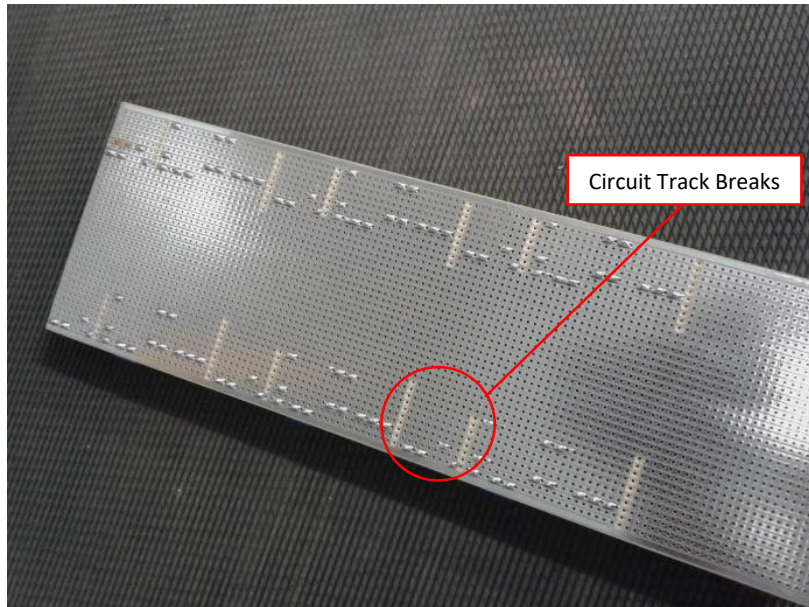


Figure 4.9: Soldered relay circuit underside with short-circuit breaks

Figure 4.10 shows the wire connections of the devices connected to the PCB terminals. The wires were labelled, with colour-coding for positive (red) and negative (black) connections. Screw terminals held the wires in place on the board, and banana plugs were attached to the opposite ends of the wires for connection to the PLC ports. The circuit diagram for these connections can be interpreted from Figure 4.21 in Section 4.5.1.

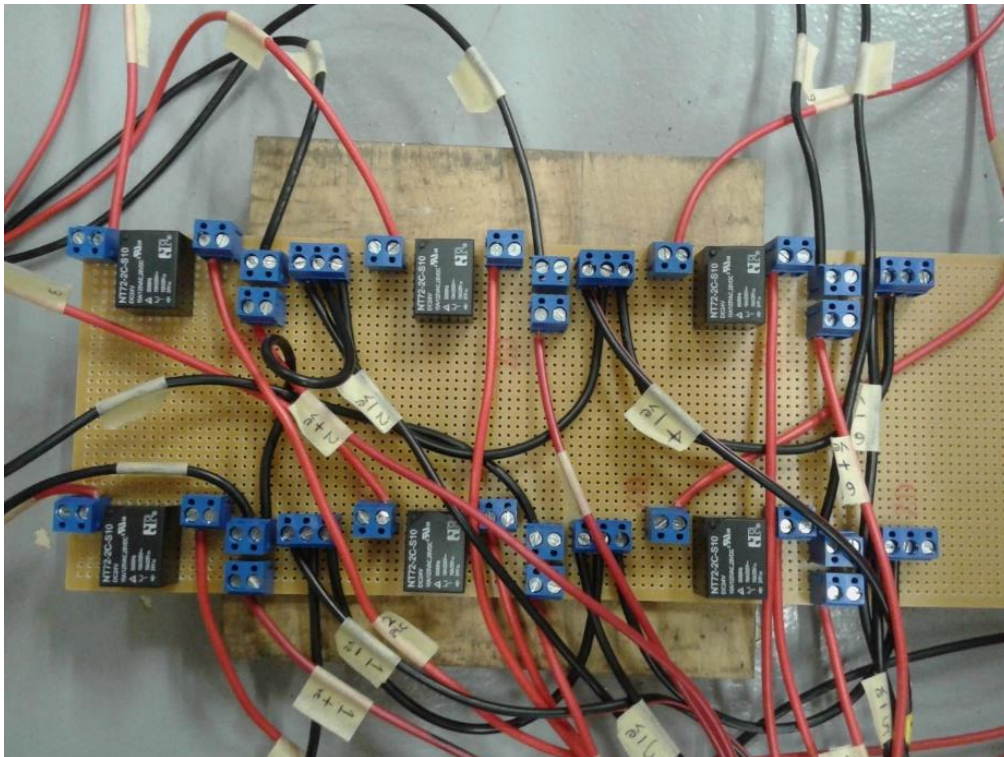


Figure 4.10: Relay circuit with wire connections

The relay circuit was mounted on a platform, with a lid to cover it; this formed the relay circuit housing. The housing was 3D-printed with PLA plastic. Figure 4.11 shows the platform being fabricated by the 3D printer.

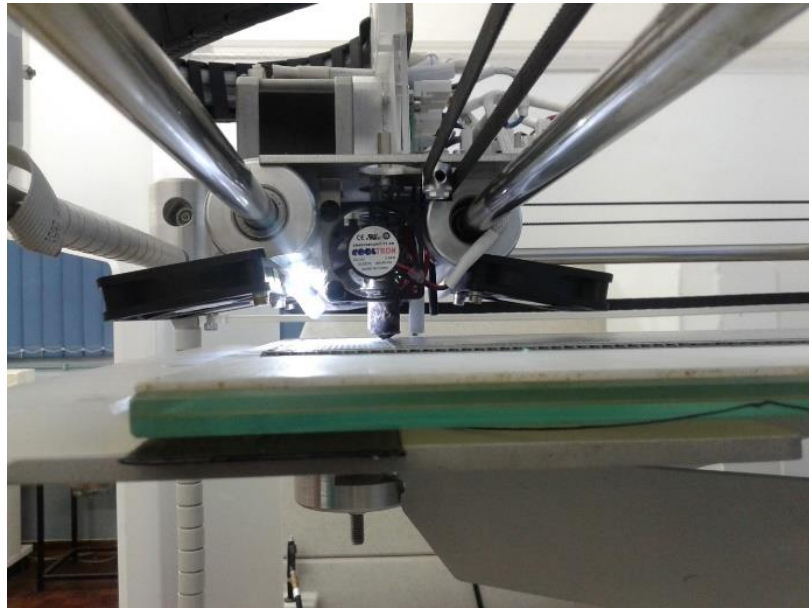


Figure 4.11: Relay circuit housing being 3D-printed

Figure 4.12 shows the relay circuit in housing and the 12 V DC PSU; both devices were mounted on a strut beam under the 'To PMS' conveyor, from where the connections to the DC motors and PLC branched out.

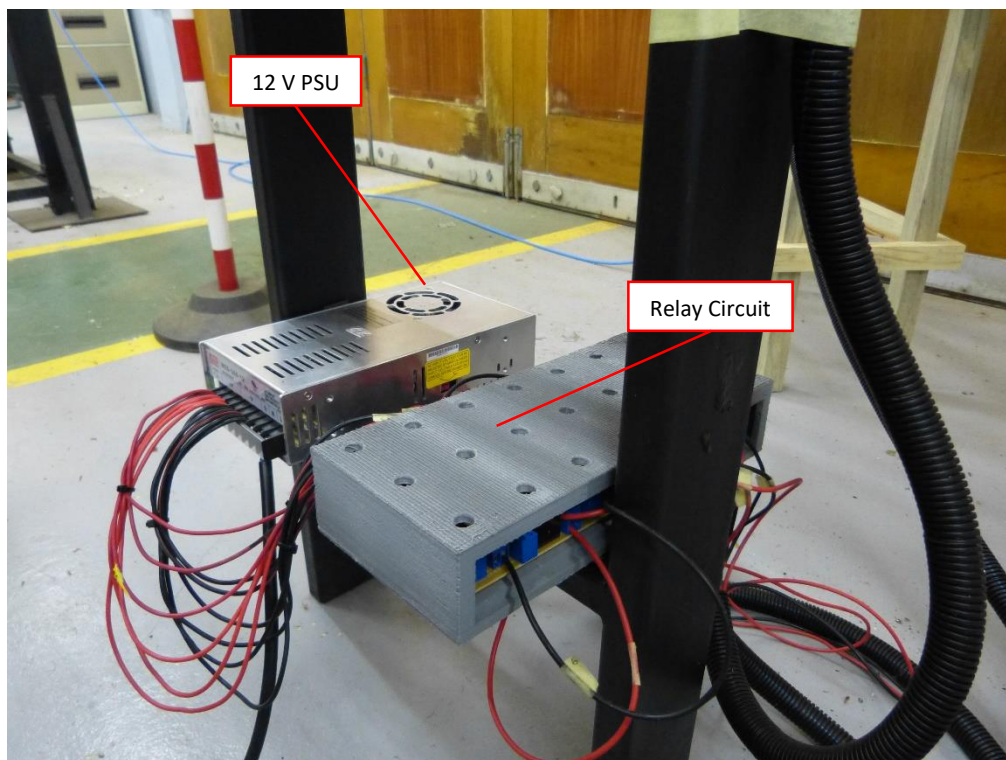


Figure 4.12: PSU (left) and relay circuit in housing (right)

4.4 Pneumatic System

The automation of the workflow of the fixture manufacturing cell necessitated a pneumatic system to drive the actuator when in Reconfiguration Mode. Reconfiguration Mode is the state of the FxMC when the operator is required to reconfigure the fixture being retrieved from the ASRS. The fixture had to be diverted away from the 'To PMS' conveyor and directed to the Reconfiguration Station. The redirection of the fixture to the Reconfiguration Station was achieved by a pneumatic actuator that pushed the pallet along the 'To Fixture Reconfiguration' conveyor branch. This section details the implementation of the pneumatic system hardware in the FxMC.

Figure 4.13 shows the pneumatic circuit for the system implemented. The system comprised a 5/2 solenoid valve operating a double-acting pneumatic cylinder (or actuator).

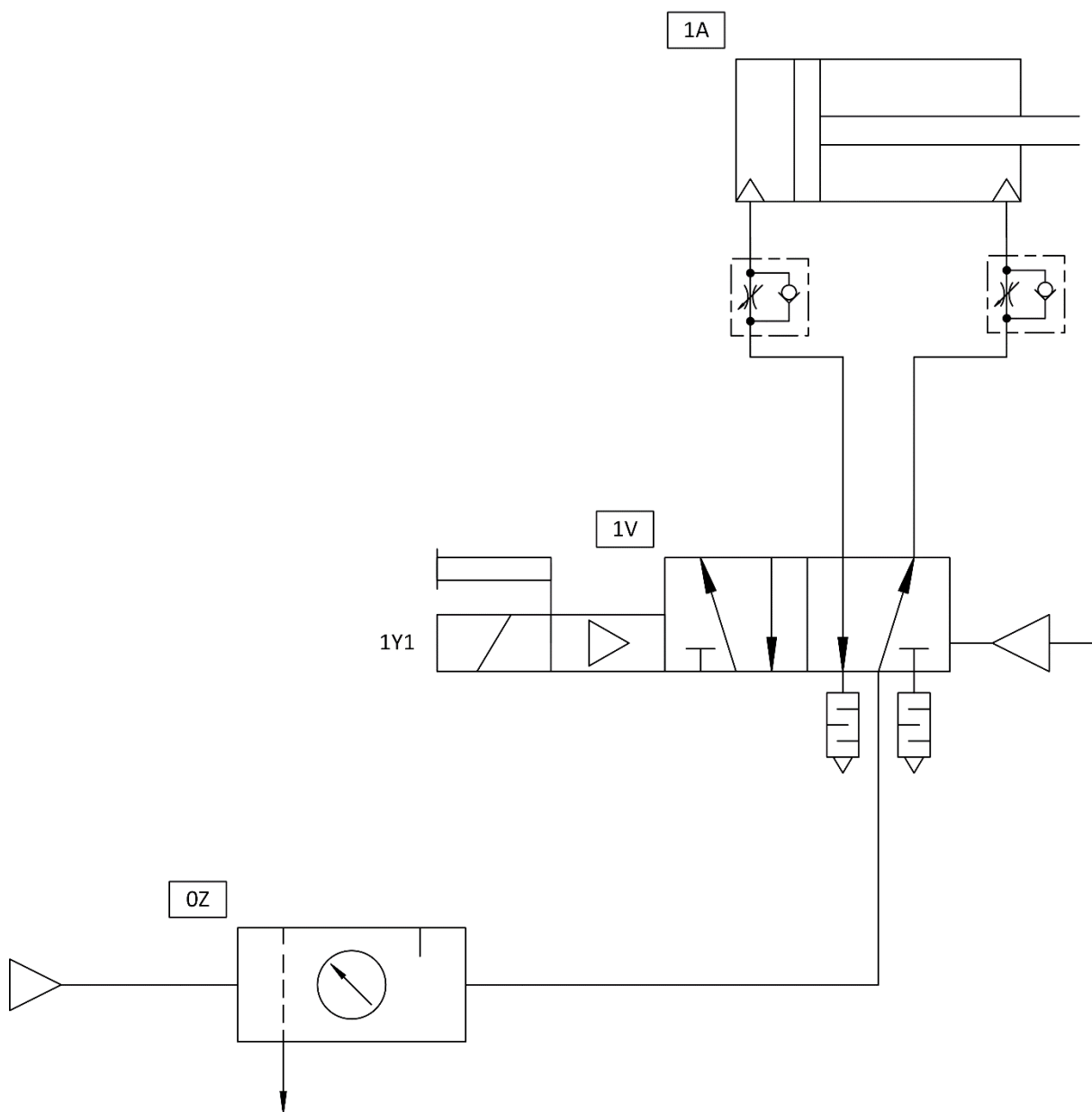


Figure 4.13: Pneumatic circuit

The service unit is denoted as '0Z' in Figure 4.13, which designates it as the power supply device for the pneumatic system [95]. An EMC® service unit was connected to the industrial air compressor outlet that was used in laboratory. An image of the service unit is shown in Figure 4.14.



Figure 4.14: FRL service unit

The service unit was of the Filter Regulator Lubricator (FRL) type. The filter cleans the compressed air; it removes contaminants via the filter element, and moisture via the drain valve. The regulator uses a rotary knob with detent to set and maintain the outgoing compressed air at the desired pressure. The lubricator moistens the air with oil, which provides lubrication to the pneumatic components used [95]. The device was rated at 0.5 – 10 bar. An adjustable horizontal fitting for a 6 mm diameter push-in fitting was already attached to the service unit outlet.

The 5/2 solenoid valve is denoted '1V' in Figure 4.13, which designates it as the control element for the actuator used; '1Y1' denotes the solenoid used to switch the valve from the 'normal state' to the 'operated state' [95]. A Festo® solenoid valve was selected (see Appendix D.5) to control the air supply to the actuator. The connections in the pneumatic system were made with 6 mm diameter polyethylene tubing; the material was selected due to its cost-effectiveness and suitability to the application; the diameter was advised by the existing fitting on the service unit.

The valve selected was of the 5/2 type, which translates to five ports and two states [95]. The five ports are branded on the device in Figure 4.15. The compressed air from the service unit was connected to Port 1 via a push-in L-fitting. Ports 2 and 4 were similarly connected to tubing that led out from the valve to the actuator. Ports 3 and 5 were exhaust outlets for returning air, with silencers (see Appendix D.6) attached to dampen the noise produced from the exhaust action. The solenoid unit is shown in Figure 4.15, with a red and black wire extending from its socket connector.

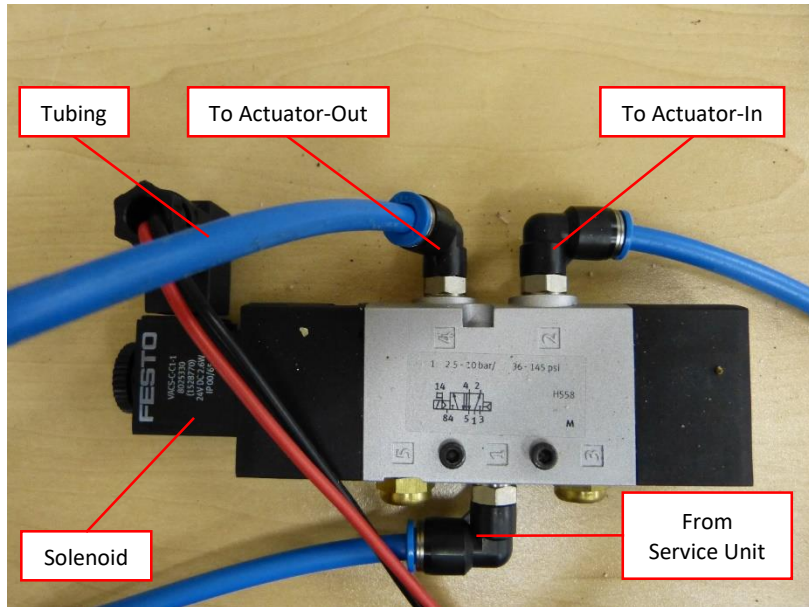


Figure 4.15: 5/2 Solenoid valve

Figure 4.16 shows the electrical circuit for the activation of the solenoid valve. The switch S1 corresponds to the PLC port to which the solenoid valve was connected; this energises the solenoid coil (K1), which drives the solenoid pin (1Y1). The solenoid pin pushes the pilot valve, which pushes the main spool; this changes the state of the valve from normal state to operated state. The process is reversed by introducing a pressure differential to return the spool to its normal state. A manual override switch is present on the unit, which can activate the pilot valve without having to energise the solenoid [95]. The entire control element (5/2 valve with auxiliary attachments), can be interpreted from the diagrammatic representation of ‘1V’ in Figure 4.13.

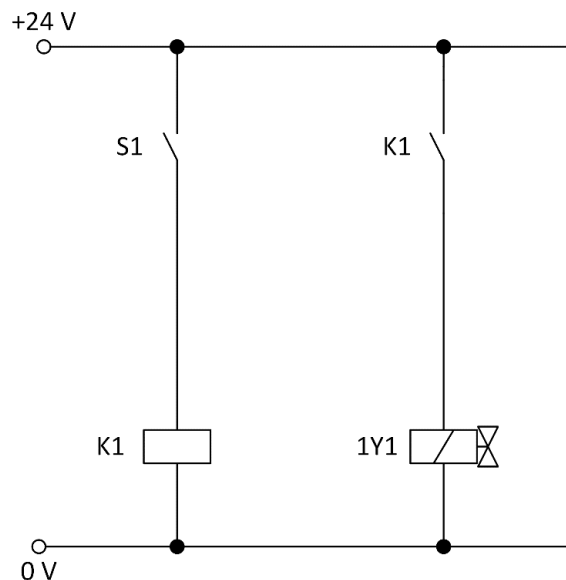


Figure 4.16: Electrical circuit diagram for pneumatic system

Figure 4.17 shows the outlet connections from the solenoid valve to the double-acting pneumatic cylinder (or actuator). The actuator was a Festo® standard cylinder (see Appendix D.3); the stroke was

specified to 500 mm to push the pallets across the ~ 400 mm wide conveyor, while the bore was the minimum available for that stroke length, since the force required to push the pallet was minimal (~ 50 N for a 5 kg part for a conservative friction coefficient of 1). The service unit was set to 2 bar for the operation of the cylinder, which would theoretically produce a force of 161 N on the advance stroke (1/3 theoretical force at 6 bar shown in Appendix D.3); this sufficiently exceeded the requirements for the operation.

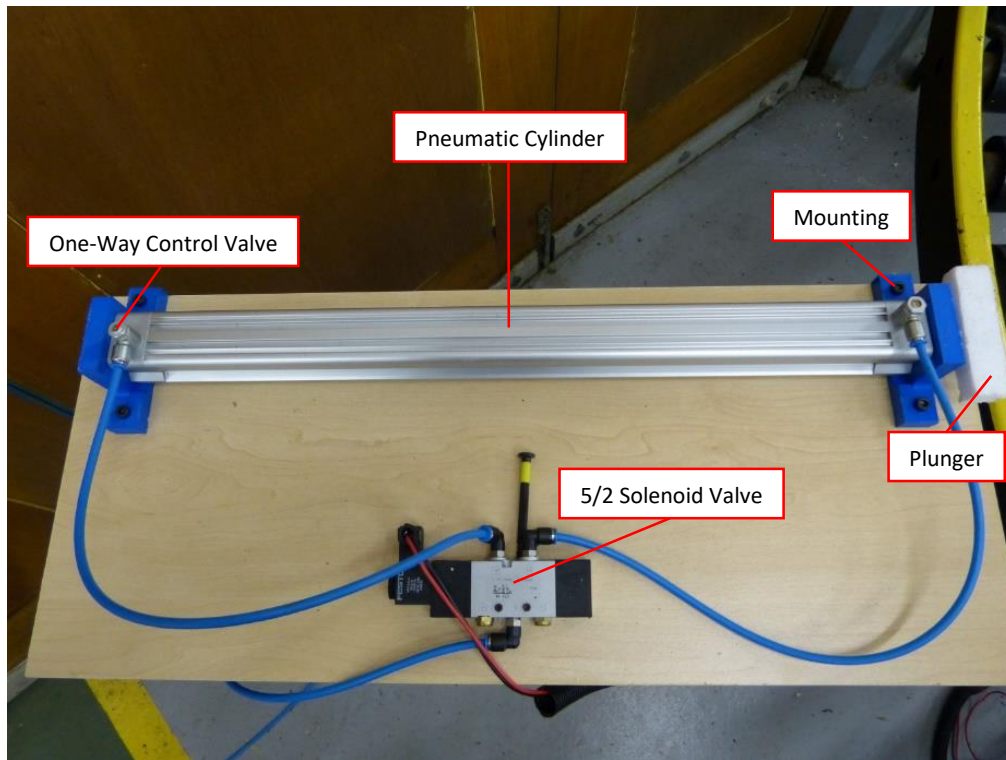


Figure 4.17: Double-acting actuator and 5/2 control valve setup

The outlet ports were each connected to a one-way control valve (see Appendix D.4) on each end of the cylinder, which restricted the exhaust flow back to the solenoid valve; this ensured that the piston rod extended and retracted against a back pressure, which reduced the speed at which this action occurred [95]. Port 4 was connected to the outstroke side, which extended the piston rod; Port 2 was connected to the instroke side, which retracted the piston rod.

Figure 4.13, in conjunction with Figure 4.15 and Figure 4.17, show how the piston rod is extended via the operated state of the solenoid valve by directing the pressurised air through Port 4, while the air already inside the cylinder is exhausted via Port 2 through the restricted control valve and out through the silencer in Port 3; retraction is then achieved by the normal state of the solenoid valve, whereby the pressurised air is directed through Port 2, while the air currently in the cylinder is similarly exhausted via Port 4 and out through Port 5. This procedure was how the pneumatic system achieved the bidirectional actuation of the piston rod through the 5/2 solenoid valve.

Figure 4.17 also shows the mounting brackets for the pneumatic cylinder; these were 3D-printed from a design that met the mounting specifications of the cylinder.

A stand was constructed to provide a platform onto which the actuator and valve were mounted. The stand was designed for a height that placed the actuator at the level required to push the pallet across the conveyor. The pneumatic system (minus the service unit) is shown in the context of the Lab FxMC in Figure 4.18.

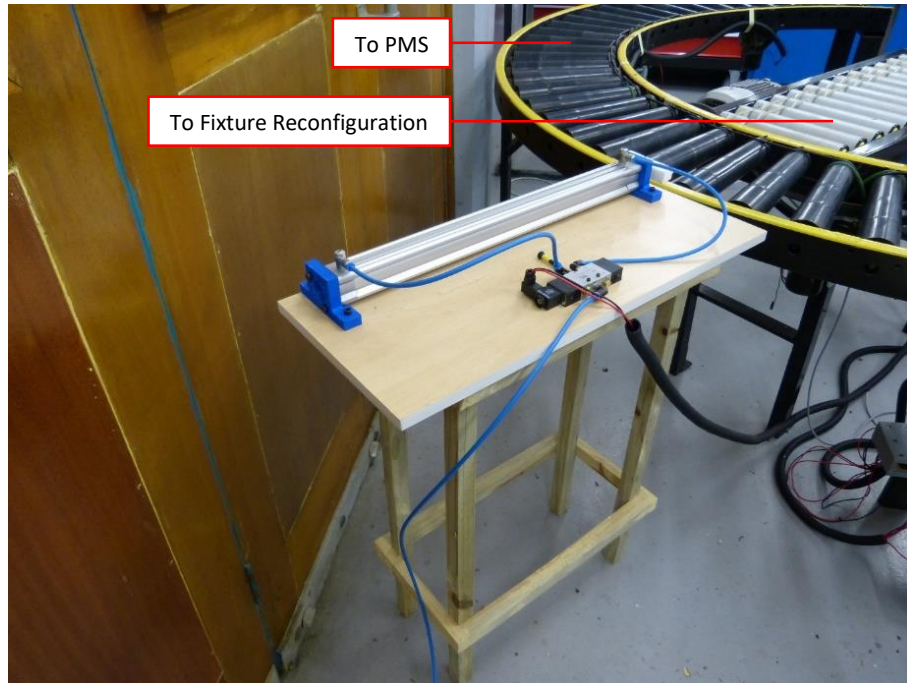


Figure 4.18: Pneumatic components

4.5 PLC

4.5.1 PLC Components

The fixture manufacturing cell workflow was automated and controlled by a programmable logic controller. The device used was a Festo® modular unit; it comprised of a CPX-CEC-C1 terminal block, which was connected to a CPX-AB input and output module, each with 16 ports. The PLC module is shown at the top of the unit in Figure 4.19. The unit was of the Festo® Didactic type, as it was equipped with banana sockets that were internally connected to the PLC input and output modules. The input and output ports for these connections are shown on the lower half of the unit in Figure 4.19.

The PLC used was a 16 Din/16Dout unit, as it comprised of 16 digital input and output ports, respectively. The ports were denoted as 0L – 7L for the left two columns of ports, and 0R – 7R for the right two columns (applied to both the input and output ports). The Didactic unit incorporated switches on the input module for manual control of input signals; four of which were used for the operation of the cell (see Section 4.5.2). Figure 4.19 shows the banana plug connectors related to the devices discussed in Sections 4.3 and 4.4. Figure 4.20 shows a schematic of which connections relate to which devices for which ports. The ports and connections in Figure 4.20 are colour-coded according the PLC ports and wires used for the connections; the input and output devices are labelled according to their names (or addresses) in the PLC code (see Section 4.5.2).

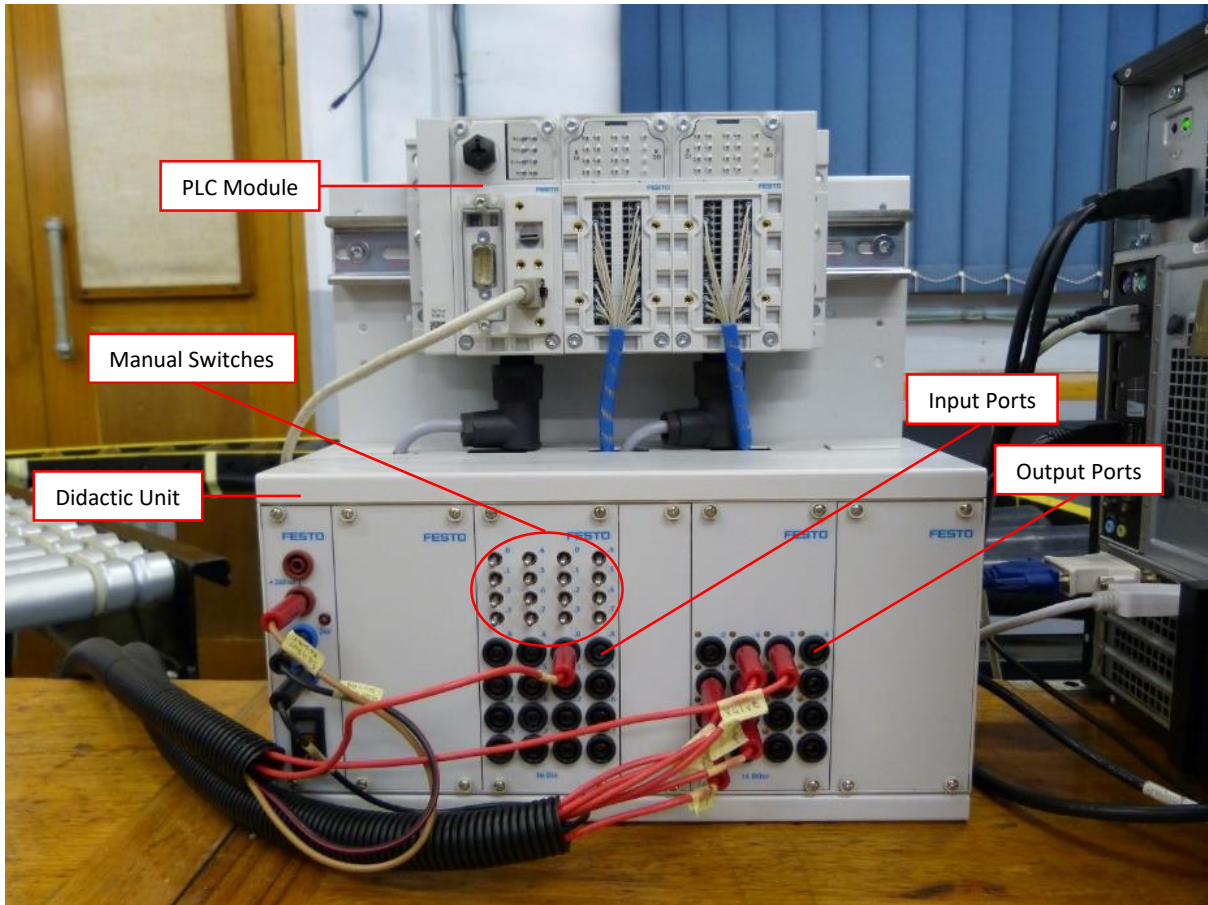


Figure 4.19: PLC module with connectors

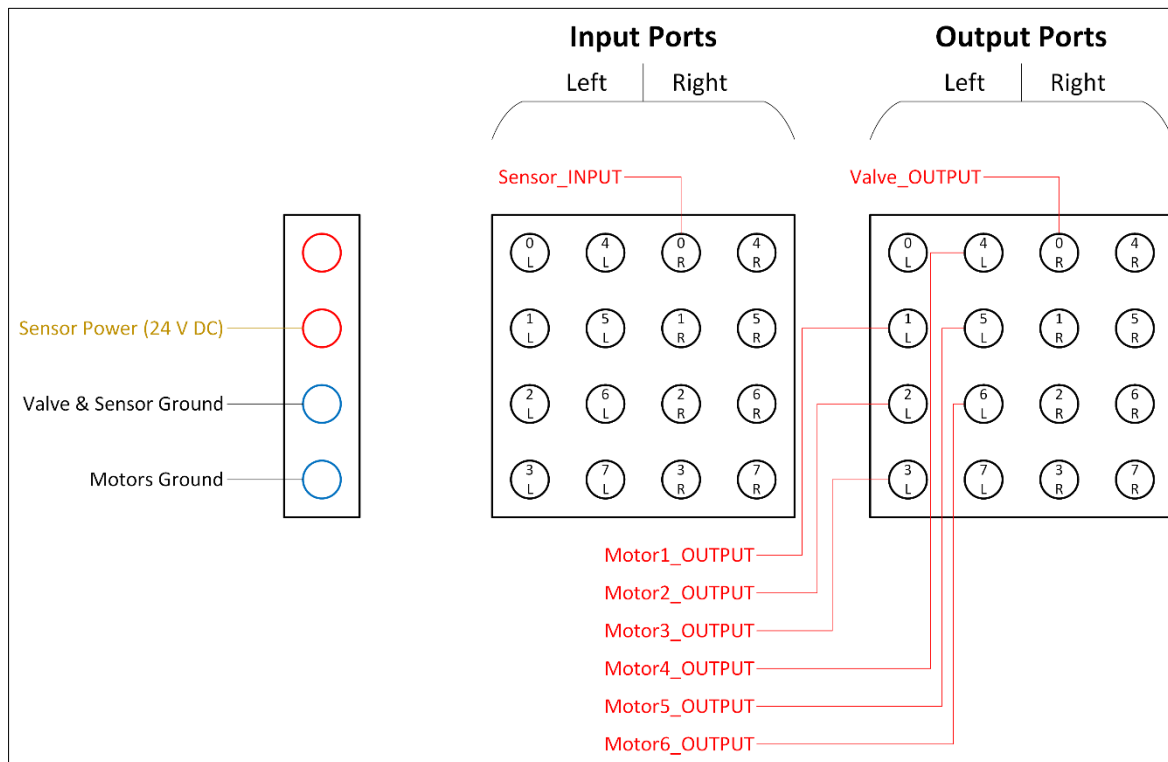


Figure 4.20: PLC connections schematic

The PLC outputs produced a signal of 24 V DC each; relays were implemented to use this signal to control the 12 V DC motors for the conveyors (see Section 4.3). The conveyor motors were independently controlled by their corresponding PLC output ports. The solenoid valve (see Section 4.4) was also independently connected to a corresponding PLC output port. Figure 4.21 shows the electrical circuit schematic for the devices connected to the PLC output module.

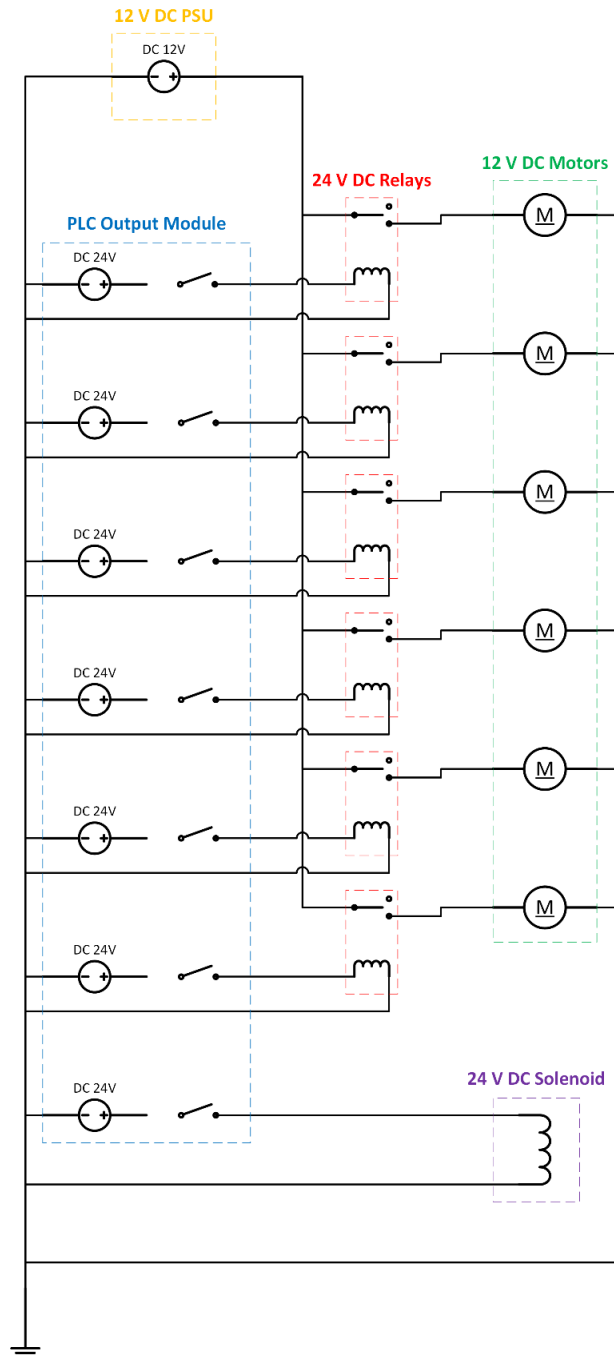


Figure 4.21: Electrical circuit schematic for PLC output devices

A Festo® diffuse light sensor (see Appendix D.7) was the only input device connected to the PLC input module; this was an optoelectronic sensor, which measured the reflection of the red output light due to an object in the path of that light. The sensor was used to notify the pneumatic cylinder of an incoming pallet that has to be pushed along the conveyor branch. The distance range for which the sensor was

activated was adjusted with an internal potentiometer. The sensor was powered from a separate port on the PLC unit, which produced a steady 24 V DC output (not usable as a control signal), as shown in Figure 4.20. The sensor is shown in Figure 4.22, where it is mounted on a 3D-printed mounting plate.

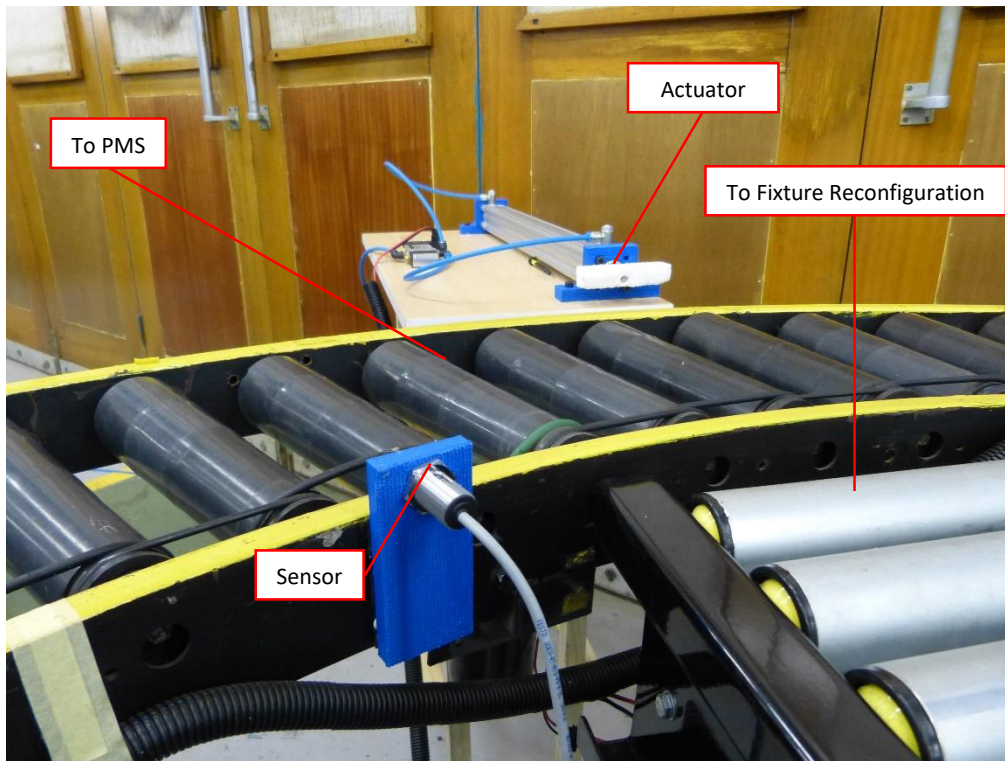


Figure 4.22: Intersection point for fixtures to be reconfigured

The other inputs employed by the PLC were via the in-built manual switches. Switches corresponding to input ports 1L – 3L (see Figure 4.20) were used to control three separate segments of the conveyor system (as shown in Figure 4.5), where ‘To PMS’ conveyor was divided into two separations (‘To Branch’ and ‘To Part’). The segments corresponding to the switches were denoted as:

- 1) **To-ASRS:** the curved conveyors leading to the input of the ASRS (as denoted in Figure 4.5).
- 2) **To-Branch:** the first half of the ‘To PMS’ conveyor in Figure 4.5; the segment before the conveyor branch ‘To Fixture Reconfiguration’.
- 3) **To-Part:** the second half of the ‘To PMS’ conveyor in Figure 4.5; the segment leading to the PMS, after the conveyor branch ‘To Fixture Reconfiguration’.

The switches for the conveyor segments were used to turn the motors on, from which a constant ‘on’ signal could be used in conjunction with the ladder logic programme (Section 4.5.2) to automate the cell. The switches were also employed as a safety feature, from which the motors could be manually turned off by the operator if necessary.

The switch corresponding to input port 0L was used for Reconfiguration Mode. Reconfiguration Mode was turned on by the operator when the retrieved fixture was required to be reconfigured; this activated the pneumatic sensor to push the pallet to the Reconfiguration Station. The mode was turned off when the retrieved fixture was already in the appropriate configuration, thus causing the actuator to ignore the diffuse light sensor signal.

4.5.2 PLC Code (Ladder Logic Programme)

The PLC was programmed for the desired operation of the fixture manufacturing cell. A ladder logic programme was developed. Ladder logic is a symbolic programming language used for PLCs, which is often likened to a relay schematic [96]. Instructions are coded as:

- XIC (Examine If Closed): Analogous to a NO relay contact; examines address, writes TRUE if 1, FALSE if 0.
- XIO (Examine If Open): Analogous to a NC relay contact; examines address, writes FALSE if 1, TRUE if 0.
- OTE (Output Energise): Analogous to a relay coil; writes 1 to address if incoming statements are TRUE (logic continuity), writes 0 to address if any incoming statements are FALSE (no logic continuity).

Addresses are the labels above the symbols in the schematic (see Figure 4.23). The addresses may denote an externally connected device or an internal input/output (referred to as an internal relay). Advanced functions exist, such as timers, which can provide more complex capabilities than the binary logic of the basic symbols. The functions are arranged in sequence, on rows referred to as rungs. The rungs are read from left to right, progressing from the first rung to the last. Reading of these rungs by the PLC is referred to as the scan cycle, which examines the state of the inputs at the start of the scan and then writes to the outputs upon completion of the scan [96].

Figure 4.23 shows the PLC programme that was developed for the cell. The programme ladder is comprised of eight rungs. The addresses corresponding to real input and output devices are cited in Figure 4.20. Other addresses are internal relays that do not correspond to real devices, i.e. they are artificial inputs and outputs.

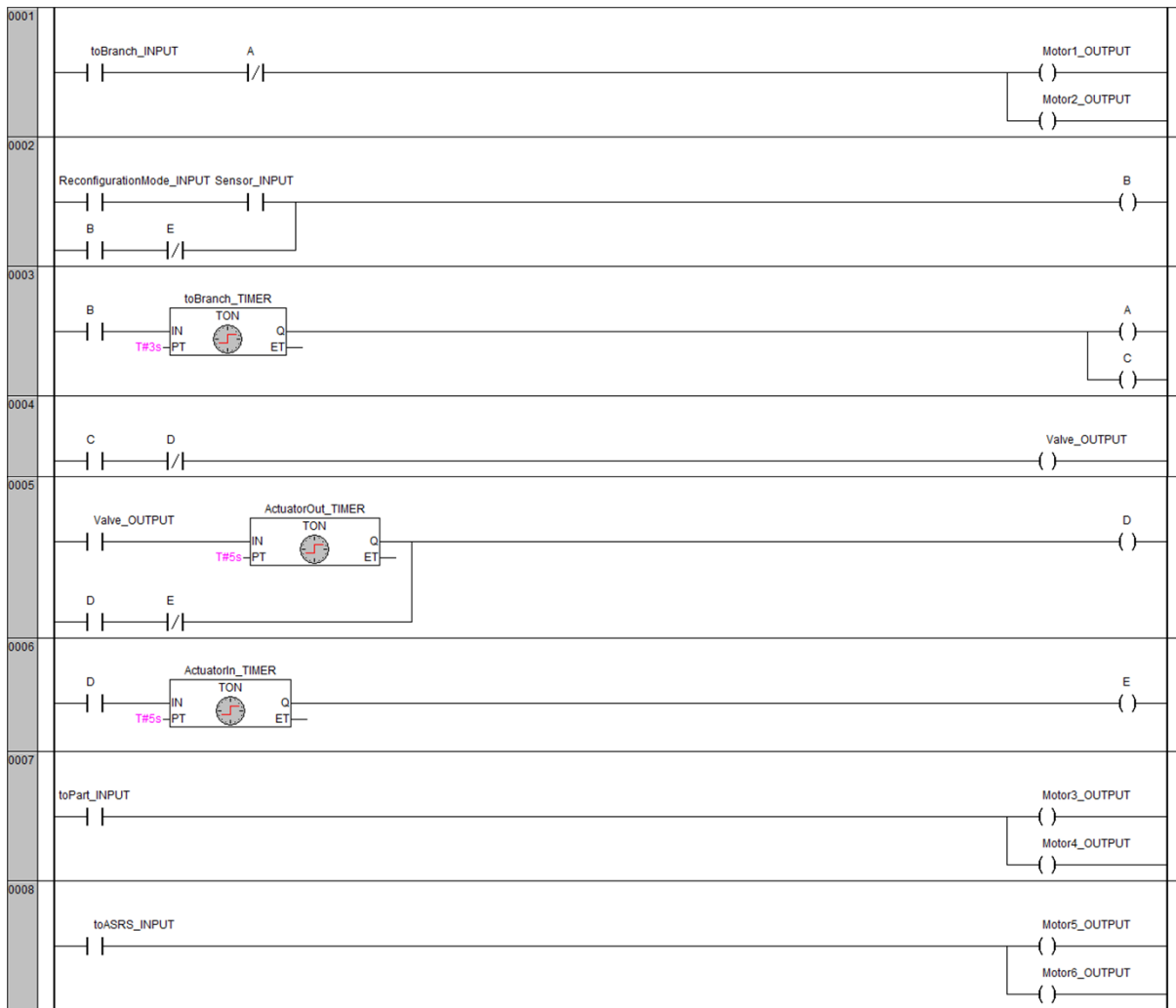


Figure 4.23: PLC programme

The programme in Figure 4.23 is shown in the Programming Mode of the PLC, i.e. logged out from the PLC. Figure 4.24 shows the PLC programme when logged into the PLC via the Ethernet connection, i.e. Run Mode. The initial state of the programme shows the XIO contacts to be TRUE (highlighted in blue).

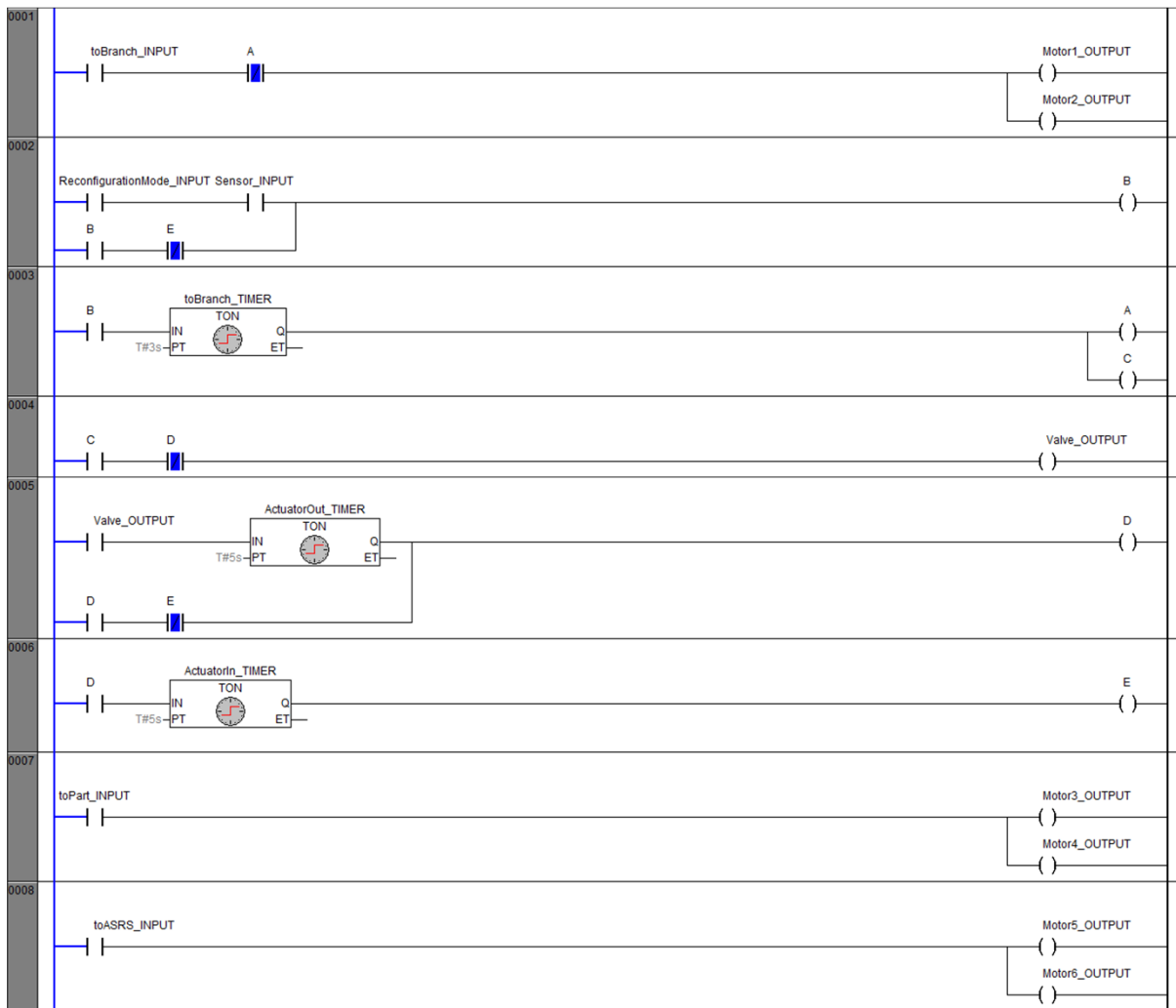


Figure 4.24: PLC programme (initial state)

Figure 4.25 shows the activation of Rung 0001. Rung 0001 is activated by the manual switch for toBranch_INPUT (Switch 1L). The switch creates a logic continuity with the XIO contact ‘A’ to energise the relays for Motors 1 and 2, i.e. the To-Branch conveyor segment.

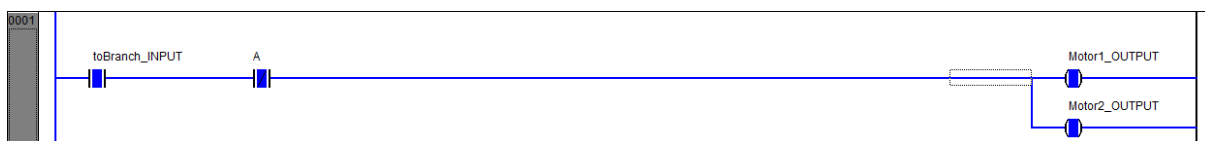


Figure 4.25: Rung 0001 activation

Figure 4.26 shows the activation of Rung 0007. Rung 0007 is activated in a similar way to Rung 0002, with the toPart_INPUT switch (Switch 2L) being turned on by the operator; this energises the relays for Motors 3 and 4, i.e. the To-Part conveyor segment.

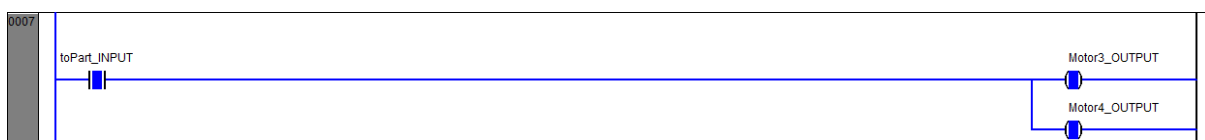


Figure 4.26: Rung 0007 activation

Figure 4.27 shows the activation of Rung 0008. Rung 0008 functions likewise to Rung 0007, except that Switch 3L is used to activate Motors 5 and 6, i.e. the To-ASRS conveyor segment.

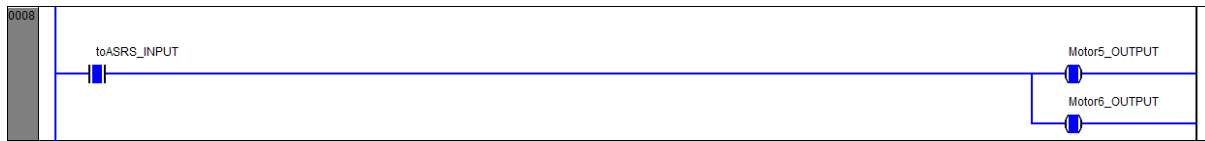


Figure 4.27: Rung 0008 activation

The operator turns on the aforementioned switches (1L to 3L) upon start-up to initiate the workflow in the fixture manufacturing cell. The pallets can travel throughout the cell in a unidirectional manner due to the activation of the conveyor motors (see Section 3.5).

Figure 4.28 shows the activation of Reconfiguration Mode via Switch 0L in Rung 0002. The logic continuity in Rung 0002 is broken due to the FALSE state of Sensor_INPUT. The cell is now ready to divert any fixture retrieved from the ASRS to the Reconfiguration Station. A parallel connection is shown on the input side of Rung 0002; this is known as a ‘seal-in rung’. A seal-in rung ensures that the activation of the OTE in the main rung due to a momentary XIC condition is kept activated even after the XIC signal has changed back to FALSE, i.e. it ‘seals in’ the activation of the OTE due to the momentary XIC [96]. This ensures that the output condition remains constant, even after the subsequent scan cycle reads the XIC signal as FALSE. This state is maintained until some other condition is met to break the logic continuity that exists due to the seal-in rung.

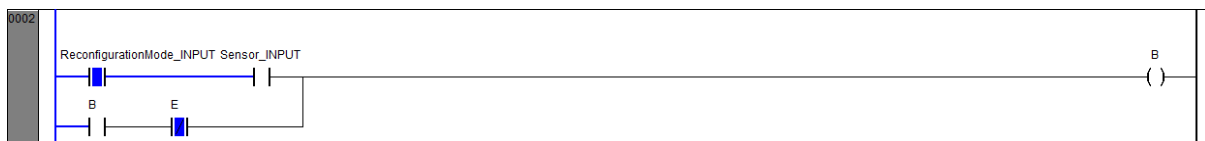


Figure 4.28: Reconfiguration Mode ready

Figure 4.29 shows the state of the ladder programme upon activation of the diffuse light sensor by a pallet in transit. This action energises the artificial OTE ‘B’, the address of which is read as an input signal to Rung 0002 in the subsequent scan cycle. The activation of B initiates the toBranch_TIMER, which delays subsequent actions in that rung for the designated 3 seconds. This is the time required for the pallet to move from the position at which it first intersects the red light beam of the sensor, to the position at which it is in front of the actuator plunger, which is aligned with the branch conveyor itself (Figure 4.22).

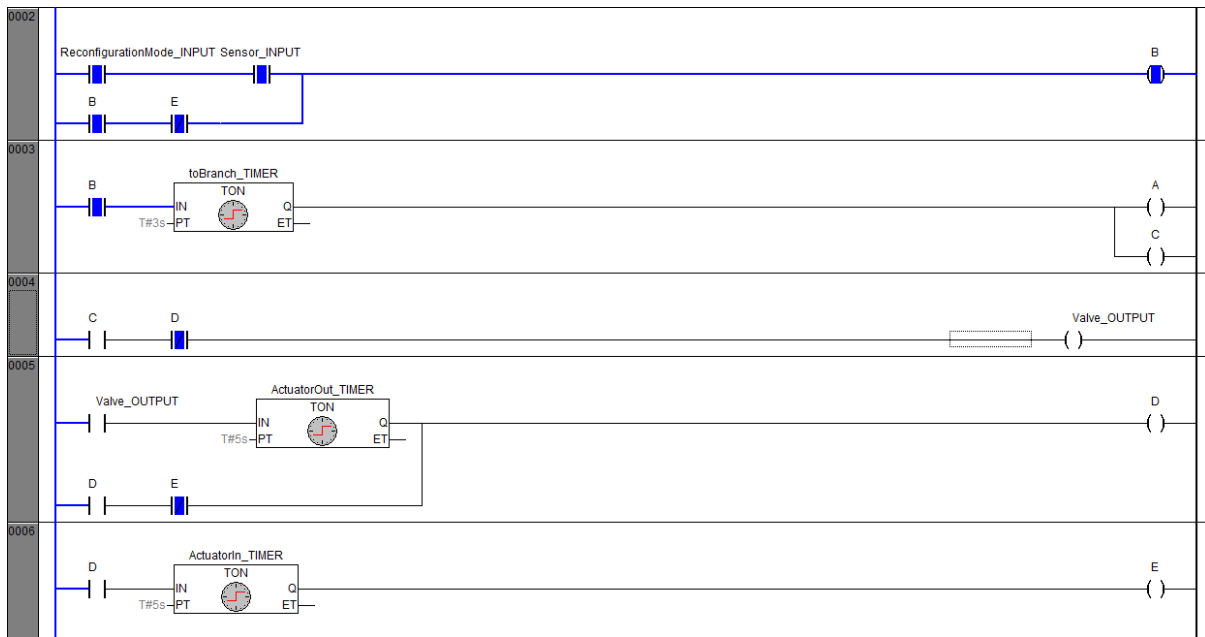


Figure 4.29: Sensor activated on Rung 0002

Figure 4.30 shows the state of the ladder programme after the toBranch_TIMER has elapsed. The sensor is no longer activated at this point, which is why the seal-in rung was required to maintain the energisation of OTE ‘B’. Logic continuity is evident in Rung 0003, which energises both artificial OTEs ‘A’ and ‘C’. The activation of OTE ‘A’ writes a 1 to the artificial XIO ‘A’ in the next scan cycle, which breaks the logic continuity of Rung 0001 (as shown in Figure 4.31). This switches Motors 1 and 2 off, which brings the pallet to a halt at the required position for the plunger; as shown in Figure 4.32. The activation of OTE ‘C’ writes a 1 to the XIC ‘C’ in Rung 0004, which completes the logic continuity required in that rung for the activation of the solenoid valve. This initiates the outstroke of the pneumatic actuator, which extends the piston rod, as shown in Figure 4.33. The real output of Valve_OUTPUT is then used as an artificial input in Rung 0005, where it initiates a timer. The timer delays further actions for 5 seconds, which is the time required for the piston rod to extend.

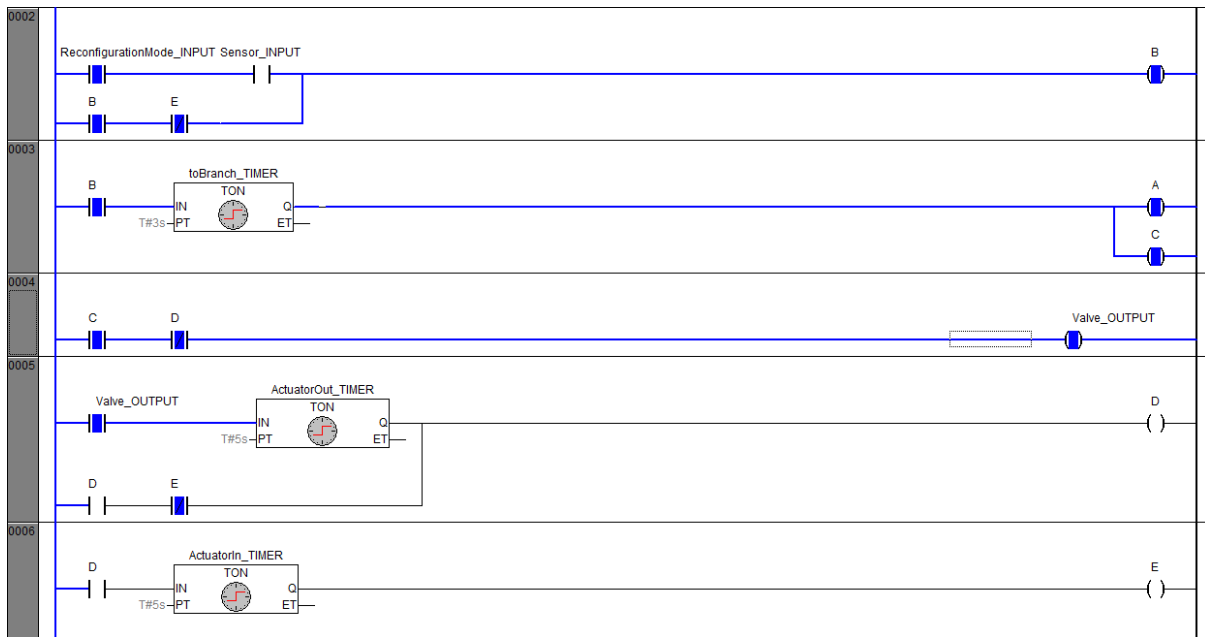


Figure 4.30: Valve activated on Rung 0005, waiting for timer to elapse

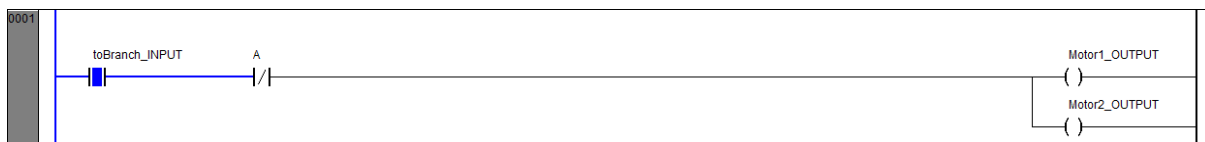


Figure 4.31: Motors 1 and 2 deactivated

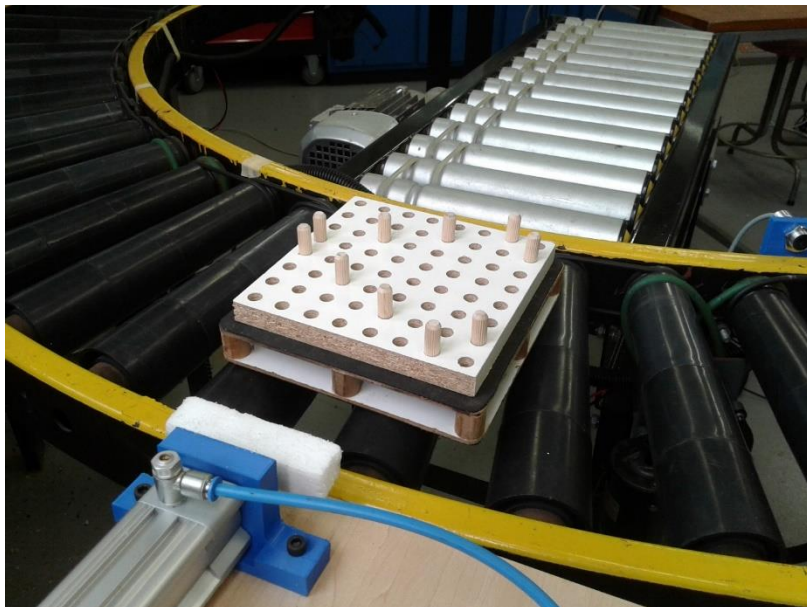


Figure 4.32: Conveyor segment paused, pallet waiting at position

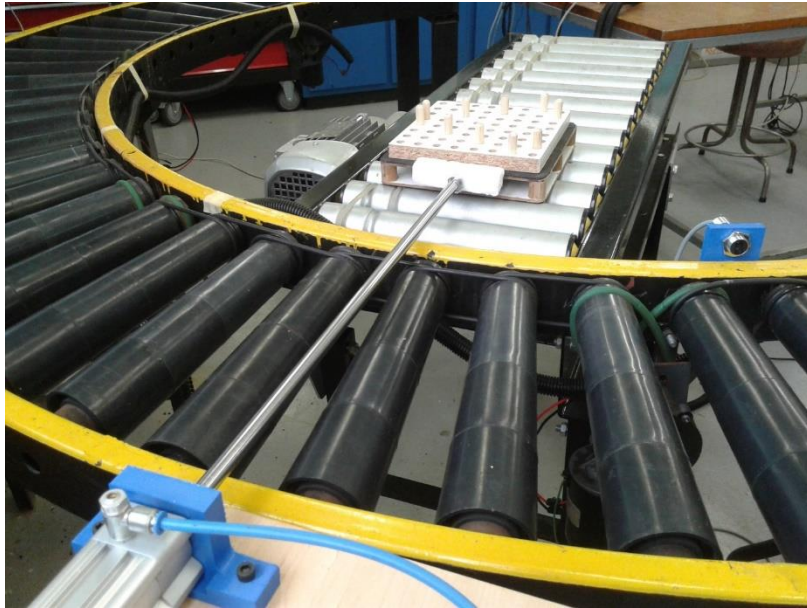


Figure 4.33: Piston rod extension to push pallet along branch conveyor

Figure 4.34 shows the state of the ladder programme after ActuatorOut_TIMER has elapsed. The artificial OTE 'D' is energised by the input signal from the valve (which is deactivated at the subsequent scan; this is why it is shown as deactivated in Figure 4.34) and sealed-in to ensure it remains energised after the valve signal is inactive, while the artificial XIC input in Rung 0008 waits for the ActuatorIn_TIMER to elapse after 5 seconds while the piston rod retracts. After the final timer has elapsed, the artificial OTE 'E' is energised, which breaks the logic continuity in both Rungs 0007 and 0002, thus de-energising both OTEs 'D' and 'B', for use in subsequent operations. De-energising OTE 'B' energises XIO 'B' in Rung 0001 in the next scan cycle, which switches Motors 1 and 2 on for continued operation of that conveyor segment. The ladder programme is now ready to undergo the same procedure when needed again.

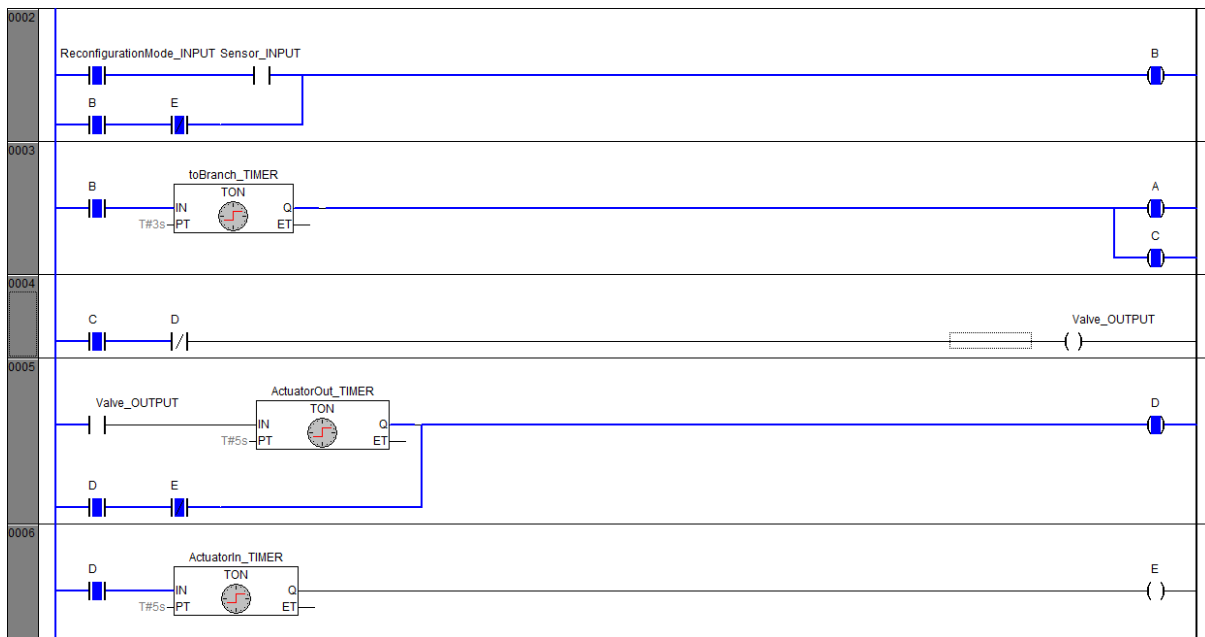


Figure 4.34: Waiting for piston rod to retract on Rung 0008 so procedure can be concluded

Figure 4.35 shows the sensor being activated when Reconfiguration Mode has been turned off by the operator. Logic continuity does not exist in the rung, which does not energise OTE 'B'. Therefore, the aforementioned procedure does not occur, thus allowing the pallet to continue along the conveyor and to the part manufacturing system without reconfiguration of the fixture.

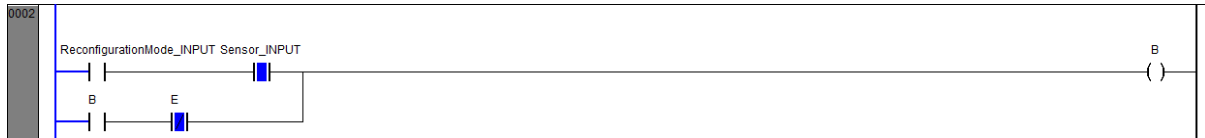


Figure 4.35: Sensor activated when Reconfiguration Mode is not activated

The straight conveyor branches were powered by three-phase motors, and were thus switched on manually; they were not controlled by the PLC programme.

4.6 Summary

This chapter summarised the practical implementation of the fixture manufacturing cell. The cell that was assembled and automated was denoted the Lab FxMC; it was based on the FxMC layout formulated in Section 3.5. The modifications to the existing infrastructure related to the cell was explained; these included mechanical refurbishments and programme code editing. The electrical/electronic system implemented for the automation of the cell conveyors was described, where a relay circuit was utilised together with a PLC unit. The pneumatic system that was employed in the cell was described. The PLC unit and its connections were described. The PLC ladder logic programme for the automation of the conveyors and pneumatic system was then explained.

Emphasis was placed on the practical execution of the tasks discussed in this chapter, as per the objectives outlined in Section 1.4. The cell is limited by the numerous variables that emanate from the manual operation of some of the tasks. Accurately verifying the performance of the scheduling method (Chapter 6) through testing in the cell would be difficult. Therefore, the Lab FxMC was primarily used as a proof-of-concept, and to conduct rough testing that could estimate the operation and transportation times for the model and simulation inputs. Testing of the FxMC with the scheduling method (Chapter 6) was predominantly conducted through the agent-based simulation (Chapter 7).

5 The Fixture Manufacturing Cell in Context of the Production Planning and Control System

5.1 Introduction

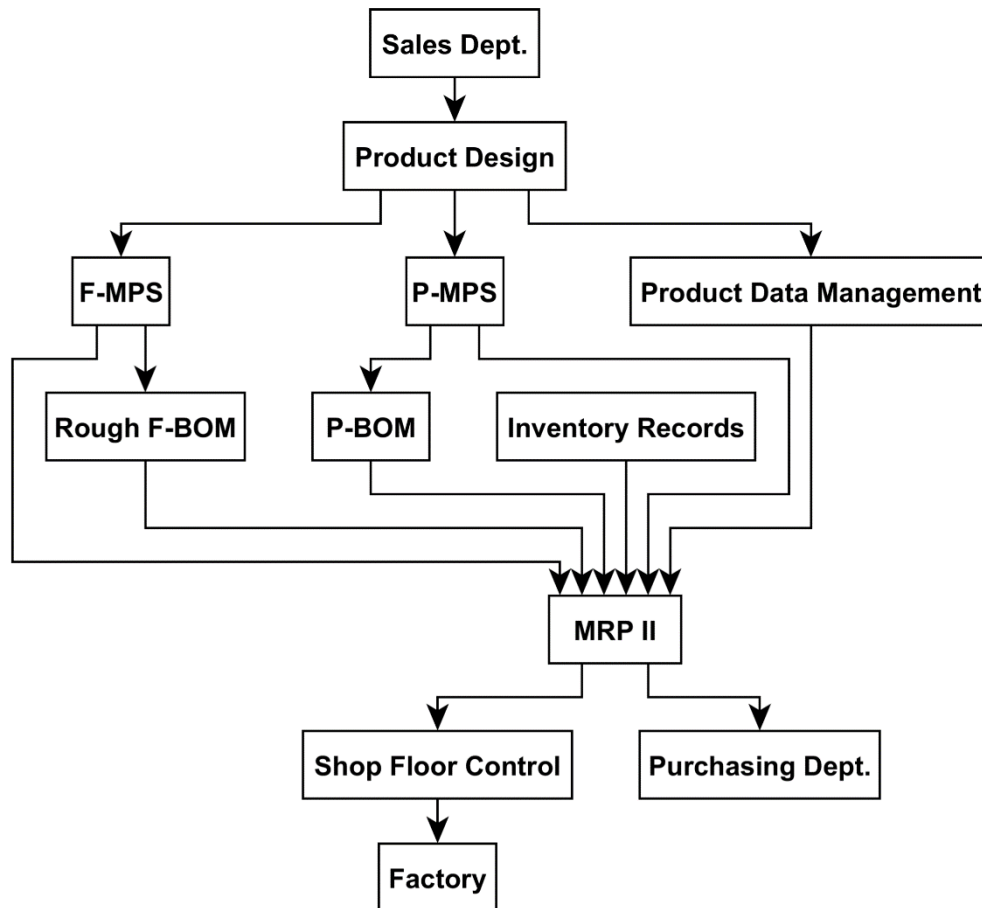
This chapter places the Fixture Manufacturing Cell (FxMC) within the context of a mass customisation production system via the Production Planning and Control (PPC) system that could be implemented. It was imperative to understand how the inputs to the scheduling method developed for the research (Chapter 6) would be obtained when integrated into the broader production system. The chapter describes the information flow through the hierarchy of the production planning system from the customer sales order to shop floor control (which is where the models employed for the scheduling method are executed).

5.2 Push/Pull Combination

Push system techniques (such as MRP II) are widely implemented in industry, but pull system techniques (such as JIT) are regarded as the focus of factories of the future [1], [6]. However, Walker and Bright [97] discovered via their Biased Decision Feedback (BDF) method, that the statistical behaviour of customer decisions in a mass customisation environment is only stabilised at high levels of WIP, thus favouring a push manufacturing system. This decision is in contradiction to theoretical literature favouring lean production methods for the future. However, the consequences of low WIP is more detrimental to the company in a competitive environment (frequently delayed product delivery and unsatisfied customers) than the higher production costs associated with buffering the higher WIP in push production systems. Furthermore, studies reviewed by Fogliatto et al. [2] revealed that customers are willing to pay fractionally more for customised products. It is for these reasons that the FxMC of the research was implemented within the context of a push production system that uses the MRP II production planning techniques described by Groover [42]. However, the operation of the shop floor, with regards to the processing of parts, was operated via pull production system techniques, as justified in Section 3.5. This ensures that the production system has adequate levels of inventory available, but the consumption of that inventory is kept lean through the workflow.

5.3 Production Planning and Control Schematic

A flow diagram of the production activities in the entire system is shown in Figure 5.1. Refer to Figure 5.3 and Figure 5.4 for subsystems of MRP II and Shop Floor Control, respectively. The following sections discuss the details of each subsystem in Figure 5.1.



Key:
F/P-MPS: Fixture/Part Master Production Schedule
F/P-BOM: Fixture/Part Bill of Materials
MRP II: Manufacturing Resource Planning

Figure 5.1: Schematic of PPC system, adapted from Groover [42]

5.4 Product Design

Figure 5.1 shows the first input to the system as the customer order at the Sales Department phase. The customer orders the product, specifying the characteristics he/she desires from it. Customisation of the product is established at this stage, which is aligned with the ‘mass customisation’ decoupling point associated with the research (Section 2.2). The order is sent to Product Design. This phase determines several variables that are to be used by the MRP II system and the scheduling method of the research. The requirements of this phase are discussed, with the understanding that the exact functionality and implementation of such systems are beyond the scope of the research, but essential to obtaining the variables that it would require in practice.

The implementation of Product Design relies on an adequate CAD/CAM system; this is an amalgamation of CAD and CAM software to integrate the design and manufacturing functions of a production system [98]. Groover [98] lists several features and functions of CAD, CAM and CAD/CAM software that are conducive to the purposes of the Product Design phase, such as the creation of manufacturing databases and the remote operation of automated machines.

5.4.1 Digital Rendering of Customer Order

Product Design receives an order from the customer via Sales Department. The design is then digitally rendered in the CAD/CAM software by a design engineer, who verifies this model with the customer specifications before dispatching it to the subsequent stages. This model contains information on the characteristics of the final product. The CAD/CAM software must convert information from this model into useful data that can be used as parameters for the manufacturing operations; this includes operation times for the manufacturing processes required to fabricate the product, and G-code for the operation of Computer Numerical Control (CNC) machines in those processes.

5.4.2 Fixture Configuration

The CAD/CAM system requires a database of basic product designs. This database stipulates the modules required, and general placement of these modules on the fixture base plate for each of the basic designs. When a customer order is received, the CAD/CAM software must infer the basic design of the product from its digital rendering and compare this to its database to retrieve the fixture configuration for such a design. This is not dissimilar to the Computer-Aided Process Planning (CAPP) system described by Groover [99], where standard process plans for part families are retrieved and attached as-is or edited for a new assembly or part design. If the shape that was ordered does not exist in the database, the CAD/CAM software must infer fixture configurations that may work to hold the part firmly. Fixture design research studies conducted in this area were listed by Esmailian et al. [6].

The next step requires the software to simulate the rotation and translation of the digital rendering of the part within the constraints of the fixture configuration. If the simulated movement is within the allowable tolerances for the process, the fixture configuration is passed.

5.4.3 Outputs of Product Design

The information contained in the CAD/CAM models are passed to the subsequent phases shown in Figure 5.1. This information is sent to Product Data Management. The CAD/CAM software must simulate the G-code programmes to be used, so that processing times can be accurately estimated for the manufacturing processes. The customer orders can be generated into a list of jobs together with their basic requirements and due dates, which form the Master Production Schedules (MPSs) in the system. The MPS can also infer the Bill of Materials (BOMs), which states the material requirements for the components of the jobs.

5.5 Inventory Management

The inventory systems of the Product Manufacturing System (PMS) were divided into:

- 1) FxMC inventory: raw materials for fixtures;
- 2) FxMC storage: storage of existing fixtures;
- 3) PMS inventory: raw material and packaging material for products;

5.5.1 FxMC Inventory

FxMC Inventory comprises of the materials for fabricating fixtures; these should include the base plate material and module material. Management of this inventory is linked to the MRP II system, with orders placed at intervals. A Reorder Point (ROP) system is used for this inventory; this is a mass production method that places an order to fully restock the inventory when the current inventory level decreases to the predefined reorder point. The reorder point is such that total depletion of the current stock is unlikely to occur before the new inventory is ordered and received [42]. A graph describing the ROP system is shown in Figure 5.2.

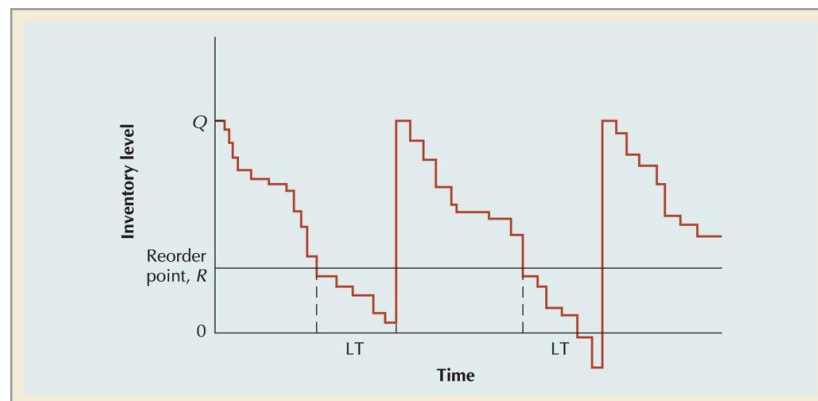


Figure 5.2: Graphical representation of reorder point system [42]

5.5.2 FxMC Storage

FxMC Storage represents the storage of completed fixtures in the FxMC. This storage is managed by an ASRS. The ASRS has a capacity limit, which limits total resource expenditure from the FxMC Inventory. This justified the simplified mass production system of ROP in the previous section. FxMC Storage is the storage pertinent to the scheduling method (Chapter 6) and practical implementation (Chapter 4).

5.5.3 PMS Inventory

PMS Inventory involves the raw material used for the product, and the packaging material thereof. The material would depend on the product type being implemented. The details of the PMS Inventory management is beyond the scope of the research. However, it is an important input (status) and output (purchasing orders) for an MRP II system implemented in an industrial PPC system.

5.6 Manufacturing Resource Planning (MRP II)

MRP II is an expansion of MRP that utilises a closed loop feedback control system to receive and dispatch data and instructions to the various subsystems in the production facility (Section 2.1). It is a push production system. The MRP II system described in Figure 5.1 is comprised of subsystems, with several inputs and outputs.

5.6.1 Inputs to MRP II

The Product Design phase (Section 5.4) generated variables that serve as direct inputs and eventual inputs to the MRP II system.

5.6.1.1 Product Data Management (PDM)

Product Data Management contains the engineering and manufacturing database relevant to the production system. PDM can be regarded as a single black box where all the engineering and manufacturing information of the received and processed jobs, and the factory floor equipment, are retrieved from, stored and updated. This includes the CAD/CAM information on the ordered products from Product Design and the process planning techniques for the production of these products. Features and limitations of the factory floor equipment, such as tool types and maximum feed rates, would also be stored in PDM.

5.6.1.2 Master Production Schedule (MPS)

The master production schedules generated from Product Design outputs are inputs to MRP II; these are separated into the Product MPS (P-MPS) and Fixture MPS (F-MPS) in Figure 5.1.

The P-MPS is a list of jobs that were dispatched from Sales Department via Product Design. The specified due dates and expected lead times for each of these jobs are listed. This provides the information on the expected PMS behaviour so that the scheduling method can make optimal decisions. The P-MPS was not created directly from Sales Department for this reason.

The F-MPS states the fixture configurations associated with each of the jobs listed in the P-MPS. The fixtures should be recycled in the system, so the F-MPS has to be separated from the P-MPS to distinguish that new resources are not expected to be consumed for every F-MPS item, but only if a new fixture is to be fabricated (unlike for the P-MPS, where every item is fabricated). Therefore, the F-MPS only associates the product type in the P-MPS with the fixture type required for it.

5.6.1.3 Bill of Materials (BOM)

The bills of materials are also separated for the products and fixtures, as they result from the respective MPSs. The Product BOM (P-BOM) lists the material used for the product and its packaging. A complicated product with assemblies and subassemblies would result in a complicated P-BOM. The Fixture BOM (F-BOM) is separated into two stages: Rough F-BOM and Final F-BOM. The input to MRP II is the Rough F-BOM, where the BOM is listed as the materials required if every fixture was fabricated, which is a worst case scenario. Thus, the Final F-BOM is shown as an output to the scheduling method which is a component of Shop Floor Control in Figure 5.4. The Rough F-BOM aids the capacity planning aspect of MRP II to ensure that there are sufficient resources for the worst case scenario.

5.6.1.4 Inventory Records

The inventory records, described in Section 5.5, are a vital input to MRP II for the MRP and capacity planning functions of the system. The inventory records relay the current status of inventory to MRP II, from which the resultant decisions on purchasing orders are made.

5.6.2 Subsystems of MRP II

In practice, MRP II is made up of more subsystems and functions than those described hereof. However, MRP and capacity planning are the only subsystems that affect the research interests. The other subsystems are summarised as Other Functions. A diagram of the subsystems of MRP II in this system is shown in Figure 5.3

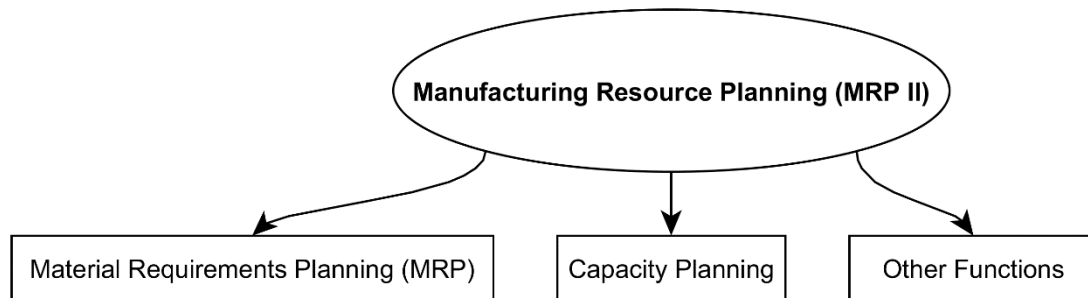


Figure 5.3: Subsystems of MRP II system from Figure 5.1

Material Requirements Planning (MRP) manages the materials in the system; this includes inventory management, which leads to purchase orders via the ROP system mentioned in Section 5.5. This subsystem can be regarded as superfluous to the research considerations, but would be considered as part of the PPC system.

Capacity Planning verifies if the solutions obtained from the MRP II subsystems are practical, based on the factory limitations and capabilities of the given facility. Capacity Planning would prevent the MRP II system from ordering more inventory than the actual storage system can handle, or prevent an infeasible MPS from scheduling an excessive number of jobs. Adjustments could be made by taking temporary measures to ensure that slight overcapacity or undercapacity from the facility is dealt with, by advising worker overtime, inventory stockpiling, etc. [42]. The details of this system was beyond the scope of the research, but its implications were noted.

5.6.3 Outputs of MRP II

The MRP II system shown in Figure 5.1 is responsible for initiating purchasing decisions for new inventory via Purchasing Department, and relaying the relevant information (prior phases included) to Shop Floor Control.

5.7 Shop Floor Control (SFC)

A traditional shop floor control system is responsible for releasing production orders to the factory, monitoring and controlling their progress through various workstations, and obtaining current information on the status these orders [42].

An identifying feature of MRP II is that it contains a SFC system, unlike MRP which treats it separately [42]. However, as this function features the scheduling method, it was decided to show Shop Floor Control separately, with a feedback loop shown to and from MRP II, in Figure 5.1.

Shop Floor Control consists of the FxMC Decisions and other functions typical of the traditional SFC system. The subsystems of the Shop Floor Control system is shown in Figure 5.4.

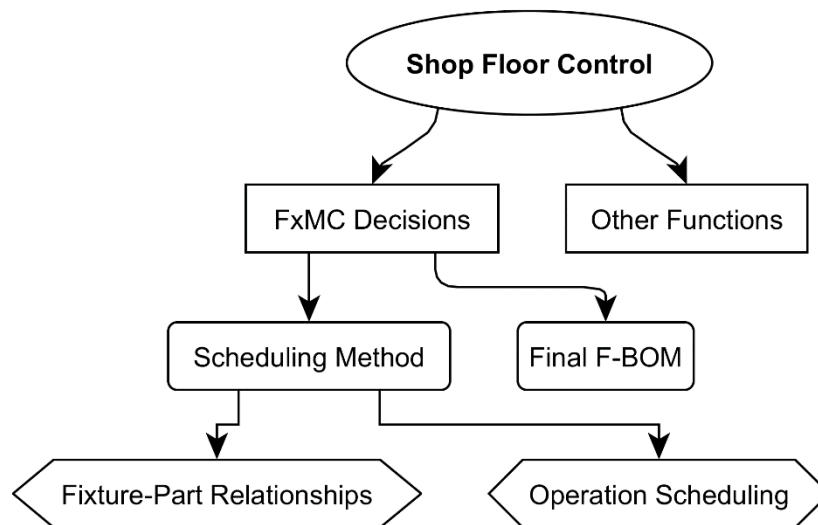


Figure 5.4: Subsystems of Shop Floor Control system from Figure 5.1

5.7.1 FxMC Decisions

The FxMC Decisions are separated into the Scheduling Method and the Final F-BOM.

The Scheduling Method requires the P-MPS and F-MPS generated from Product Design. These schedules should list the jobs, associated fixtures and estimated operation times for both fixture reconfigurations and processing of parts. Lead times for fabricating new fixtures should also be included if necessary. The Scheduling Method must handle both Fixture-Part Relationships and Operation Scheduling. The details of the solution to these aspects are discussed in Chapter 6 as a significant contribution of the research.

The results from the Scheduling Method should assist with generating the PMS and FxMC orders, which should be automatically handled by the SFC system. The results should also infer the Final F-BOM (Section 5.6.1.3). With the FxMC behaviour for the job list known, the material requirements can be accurately modelled. This information is useful to the MRP system for cross-referencing with capacity planning to eradicate prior concerns due to the worst case scenario of the Rough F-BOM.

5.7.2 Other Functions

Shop Floor Control is also responsible for activities such as actual machine operation and data collection of the production processes and their progress. This information and functionality is necessary for operation of the factory floor equipment.

5.8 Summary

This chapter discussed the production planning and control for a mass customisation system that would incorporate the fixture manufacturing cell. Reference was made to the subsystems, and the flow of information from the customer order to the execution of jobs utilising the reconfigurable fixtures. A push system was assumed for an MRP II system to be implemented, where the advantages of having high levels of inventory available for the manufacturing system was justified. However, the shop floor functionality should exhibit pull system characteristics, as described in Section 3.5. The chapter described the flow of information from the sales department to shop floor control (which is where the scheduling method of the FxMC is integrated). This information flow describes how the scheduling method could obtain the data required for it to make the optimal decisions for fixture management in-tandem with processing of parts. This chapter provided a broad context of the FxMC management in relation to a MC production system. Chapter 6 then focuses on the exact functionality of the FxMC through the scheduling method developed for the research.

6 Scheduling Method

6.1 Introduction

This chapter presents the optimal scheduling method developed for the fixture manufacturing cell (as per the fourth objective in Section 1.4). The method was separated into three stages: clustering, intracluster sequencing, and final scheduling. The chapter lists the basic optimisation requirements expected from the method. The formulation of the method in relation to the production system described in Figure 3.5 is discussed, together with justification for the three-stage structure. The three stages are summarised, before the method notation and assumptions are defined. Individual sections are dedicated to describing the mechanisms of each of the three stages thereafter. Finally, a more computationally efficient heuristic is presented for the third stage.

6.2 Optimisation Requirements

The optimisation method was required to optimise:

- The assignment of parts to fixtures;
- The scheduling of both fixture and part operations.

These were the general outcomes expected from the method, as stated in Section 5.7.1 and Figure 5.4.

6.3 Formulation

The research aimed to optimise the management of reconfigurable fixtures in a customer-driven manufacturing system. The production system representation upon which the research was conducted (Section 3.6 and Figure 3.5) revealed that the workflow exhibited JIT characteristics. The parameter for which delays were caused was shown to be the idle time produced from waiting for either Cell 1 or Cell 2 to complete its respective operation. The idle condition created a bottleneck in the system, whereby workflow was halted until both cells were available. Improvements in the manufacturing system were made by minimising these delays through synchronising the operations scheduled in both cells, creating a more time-efficient and cost-effective system. Therefore, the performance metric that was chosen to be minimised was the total idle time in the system. This would implicitly minimise the total makespan for a given job list.

Table 6.1 displays a diagrammatic table that illustrates the workflow with respect to the relationship between Cell 1 and Cell 2, where fixture-part mappings are represented as alphabets. Fixture-part mappings represent the part to be processed and the associated fixture base plate on which the pin configuration is assembled. The fixture from fixture-part mapping 'A' is reconfigured for its part in Cell 1 in the first time period; Cell 2 is empty in this period. The reconfigured fixture is then sent to Cell 2, while the fixture for fixture-part mapping 'B' enters Cell 1, both for the second time period. During this time period, the part for fixture-part mapping 'A' is processed in Cell 2 on its fixture, while the fixture for fixture-part mapping 'B' is being reconfigured for its part in Cell 1. Once both cells have completed their respective operations, the part for fixture-part mapping 'A' is dispatched, and its fixture

is released and returned for recirculation; while the newly reconfigured fixture continues to Cell 2 for the next time period, where its part is processed while the fixture for fixture-part mapping ‘C’ is being reconfigured in Cell 1. This procedure continues until the final fixture-part mapping (‘D’ in this case) is processed in Cell 2; while Cell 1 is empty, having reconfigured its last fixture for that job list in the previous time period.

Table 6.1: Workflow table with alphabet representation









Time Period	Cell 1		Cell 2	
	Fixture Reconfiguration		Part Process with Fixture	
1	A			
2	B		A	
3	C		B	
4	D		C	
5			D	

Table 6.1 also shows how the tasks for a job list are decomposed into time periods. The time period durations vary, depending on which of the two operations is the more time-consuming. The algebraic difference between the longer operation time and the other, shorter, operation time is the idle time for that time period. Therefore, the absolute time difference between the respective operation times in Cell 1 and Cell 2 (denoted fixture reconfiguration time and part processing time, respectively) for every time period can be accumulated to produce the total idle time for the job list. Hence, the objective of the optimisation method was to minimise the total idle time by finding the best pairs of fixture-part mappings to be operated on in the same time period. This would insinuate fixture-part mappings where the part processing time of the previously reconfigured mapping, and the fixture reconfiguration time of the currently reconfigured mapping, are of similar duration. However, as the objective was to minimise the total idle time, a greedy approach may not necessarily yield the optimal solution. Hence, the objective was not to minimise each individual idle time, but the total of these individual times.

The idle time calculation can be represented by the simplified Equation (6.1) below:

$$Idle\ time = |\tau * Y - \rho * X| \tag{6.1}$$

Where:

- τ = part processing time (s); a parameter
- Y = fixture-part mapping in Cell 2; a decision variable
- ρ = fixture reconfiguration time (s); a parameter
- X = fixture-part mapping in Cell 1; a decision variable

The problem was investigated with the intention of combining the two optimisation requirements into a single model. However, the complication of the two distinct requirements meant that such a model would be quadratic, as the fixture reconfiguration time would be a variable depending on which configuration that fixture is reconfigured from. A model of this type would necessitate a polynomial of a high degree in the objective function, which would significantly increase the computational expense of implementing and solving the problem [52]. Therefore, it was decided that the optimisation method should be formulated such that the assignment of fixtures to parts, and the scheduling of the resultant fixture-part mappings, should be separated.

The updated approach would have divided the method into two stages, whereby a clustering algorithm would be used to group the fixtures based on some measurement of comparison, from which the reconfiguration time would be a determinable parameter for the idle times to be calculated and minimised. An appropriate clustering algorithm for the model was to be determined; techniques from the group technology paradigm were investigated, as mentioned in Section 2.5. However, the sequencing of assignments in cellular manufacturing systems are predetermined from the process planning route sheet. The sequence in which fixture reconfigurations were performed in the research impacted the time expenditure, so the clustering techniques from GT could not be implemented as-is. This produced a further avenue for optimisation that was required to develop the schedule. Therefore, the optimal scheduling method was finally separated into three stages, as discussed in the following sections.

6.4 Method Overview

The scheduling method developed consisted of the following three stages:

- 1) **Clustering Stage (Stage I):** Groups similar parts (based on pin configurations) and assigns them to the same fixture (producing fixture-part mappings). Each group of parts is assigned to its own fixture, where the number of groups depends on the number of fixtures available. This maximises the similarity among parts in the same group.
- 2) **Intracluster Sequencing Stage (Stage II):** Sequences the parts that belong to the same group (i.e. defines the reconfiguration order within the group) such that the dissimilarity between successive parts is minimised. This reduces the extent of reconfiguration required on a single fixture, which implicitly minimises the operation times for fixture reconfigurations in that group.
- 3) **Final Scheduling Stage (Stage III):** Schedules pairs of fixture-part mappings in Cell 1 and Cell 2 with the objective of synchronising fixture reconfiguration and part processing operation times. This reduces the total idle time in the system, improving total makespan by implication.

6.5 Method Notation

The notation used to denote the scheduling method variables and parameters were as follows [100]:

$p; p \in P, P = \{1, \dots, n\}$ P is the set of parts to be processed; p is an index of the ordered set P .

$q; q \in Q, Q = \{1, \dots, m\}$ Q is the set of fixtures available; q is an index of the ordered set Q .

$i; i \in I, I = \{1, \dots, m\}$ I is the set of i , where I is a set of sets that holds all p - q mappings between sets P and Q ; i denotes a set and is also an index of the ordered set I .

$\hat{i}; \hat{i} \in I, I = \{1, \dots, m\}$ \hat{i} is an alternate index of the ordered set I .

$j; j \in i, i = \{1, \dots, i \}$	i is the set of p - q mappings corresponding specifically to fixture q , j is an index of the unordered set i ; j denotes a part p that is mapped to the fixture q .
$\hat{j}; \hat{j} \in i, i = \{1, \dots, i \}$	\hat{j} is an alternate index of the unordered set i .
$k; k \in K, K = \{1, \dots, n+1\}$	K is the set of time periods in which parts or fixtures are processed or reconfigured, respectively; k is an index of the ordered set K .
$\check{k}; \check{k} \in K, K = \{1, \dots, n+1\}$	\check{k} is an alternate index of the ordered set K .
$\rho_{\hat{i}\hat{j}}$	Fixture reconfiguration time; time for fixture \hat{i} to be reconfigured to pin configuration corresponding to $\hat{j} \in \hat{i}$ from pin configuration corresponding to $(\hat{j}-1) \in \hat{i}$ (implicitly), i.e. subsequent reconfiguration for fixture \hat{i} ; $\rho_{\hat{i}\hat{j}}$ is a parameter.
τ_{ij}	Part processing time; time for part p corresponding to fixture-part mapping $j \in i$ to be processed; τ_{ij} is a parameter.
X_{ijk}	A binary decision variable; $X_{ijk} = 1$ if fixture i is reconfigured for the fixture-part mapping $j \in i$ in time period k , $X_{ijk} = 0$ otherwise.
$Y_{i\hat{i}j\check{k}k}$	A slack variable; $Y_{i\hat{i}j\check{k}k} = 1$ if fixture-part mapping $j \in i$ that was reconfigured in time period k is processed in time period $\check{k} = k+1$ whilst fixture-part mapping $\hat{j} \in \hat{i}$ is synchronously being reconfigured in time period \check{k} , $Y_{i\hat{i}j\check{k}k} = 0$ otherwise.
$Z_{i\hat{i}j\check{k}k}$	A slack variable; $Z_{i\hat{i}j\check{k}k}$ is equal to the absolute time difference between part processing time τ_{ij} for fixture-part mapping $j \in i$ reconfigured in time period k , and fixture reconfiguration time $\rho_{\hat{i}\hat{j}}$ for fixture-part mapping $\hat{j} \in \hat{i}$ that is being reconfigured in time period $\check{k} = k+1$, i.e. the idle time for every time period where two operations are synchronous.

A sample problem is used to aid explanations of the method mechanisms. This was a 12-part/4-fixture problem (as used for the sample problem in Chapter 8). The answers to this problem are used to illustrate some of the notations above. These are summarised in Table 6.2.

Table 6.2: Method notation sample explanations

Notation	Sample problem answer	Explanation
p	[1 2 3 4 5 6 7 8 9 10 11 12]	12 parts to be scheduled.
q	[1 2 3 4]	4 fixtures available.

Table 6.2: Method notation sample explanations (continued...)

I	$\begin{bmatrix} 3 & 1 & 5 & 0 \\ 6 & 4 & 11 & 0 \\ 9 & 10 & 0 & 0 \\ 8 & 7 & 2 & 12 \end{bmatrix}$	Fixtures as rows, parts as elements; zeroes used to complete matrix where fixture is recycled fewer times than others.
$i = 2$	$[6 \ 4 \ 11]$	Part sequence for second fixture, derived from I .
$j \in i = 2 \in 2$	4	Part 4, which is sequenced as second element on Fixture 2.

6.6 Method Assumptions

The assumptions adopted for the development of the method were as follows [100]:

- Fixture reconfiguration times (ρ_{ij}) and part processing times (τ_{ij}) are predetermined parameters.
- There are fewer fixtures than parts; $m > n$.
- Cell 1 and Cell 2 exhibit a unit workflow policy; only one job is operated on at a time per cell.
- Just-in-time workflow characteristics; a job does not exit Cell 1 until Cell 2 is available, Cell 1 does not start a new job until the previous job has exited the cell.
- The fixture reconfigured in Cell 1 in time period k is used to process the part assigned to it in Cell 2 in the next time period, i.e. $\check{k} = k+1$ always.
- Once a part or fixture is assigned to a period k , it is processed or reconfigured, respectively, without interruption.
- Transportation time is negligible.
- The required number of fixtures are already manufactured and stored, so that only reconfigurations are required.
- The time taken to insert a pin module is equal to the time taken to remove a pin module, i.e. reconfigurations are bidirectional.

6.7 Clustering (Stage I)

The Clustering stage creates the fixture-part mappings by assigning similar parts to a shared fixture. The similarity of these parts are defined by the pin configurations that are assembled on the base plate of that fixture for the purpose of processing said part. The extent of reconfiguration required for conversion from one pin configuration to the next, depends on the pins that are removed and/or inserted. The research regarded the actions of both removing a pin and inserting a pin as a negative influence on the operation time that would be required for that fixture reconfiguration operation.

The Clustering stage aims to group parts into families where the extent of reconfigurations required are minimised amongst the family members. The k-means clustering algorithm is used to achieve this. The criteria by which the clustering is performed is a binary dissimilarity measure that quantifies the comparative relationships between all pin configurations. The measure used is non-Euclidean, so the distance matrix generated from the comparisons is non-metric. As such, Multidimensional Scaling (MDS) is used to scale the non-metric distance matrix into two dimensions, so that the k-means

clustering algorithm can be applied. The procedure for Stage I is discussed together with an illustrative example on how the data is represented and manipulated.

The pin configurations are assembled on the fixture base plate array, as described in Section 3.4. The dimensions of the array was specified as 8×8 holes. Pin configurations are assembled on the base plate array in an arrangement that conforms to the shape of the two-dimensional part to be processed on the fixture. Figure 6.1 displays the pin configuration that would be implemented for a large square-shaped part, with shaded holes representing the pin placements.

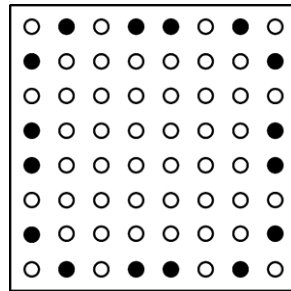


Figure 6.1: Pin configuration for large square-shaped part

The first step of Stage I is to represent this pin configuration as an 8×8 binary matrix. A ‘1’ represents the presence of a pin in a hole at that location, while a ‘0’ similarly represents the absence of a pin. Binary data representation was considered favourable for the application, due to the efficient computation associated with it [101]. The binary matrix representation of the pin configuration in Figure 6.1 is shown in Figure 6.2.

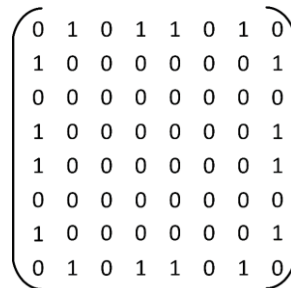


Figure 6.2: Binary matrix representation for Figure 6.1

The pin configuration for a large triangular-shaped part is shown on the left of Figure 6.3, with its binary matrix representation shown on the right. The fixture used for the large square-shaped part is to be reconfigured for this part thereafter.

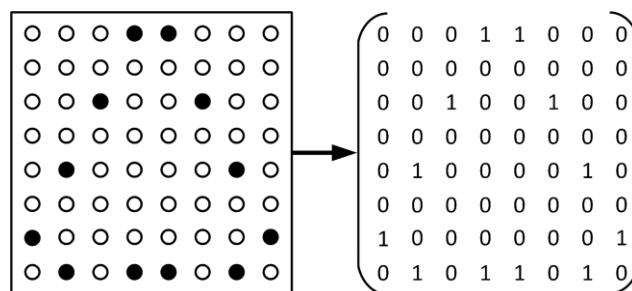


Figure 6.3: Pin configuration to binary matrix representation for large triangular-shaped part

The binary dissimilarity measure used was based on the Rand index for similarity, as shown in Equation (6.2) below [102].

$$S = \frac{a + d}{a + b + c + d} \quad (6.2)$$

Where:

S = similarity value

a = number of positive matches (1 to 1)

b = number of positive-to-negative mismatches (1 to 0)

c = number of negative-to-positive mismatches (0 to 1)

d = number of negative matches (0 to 0)

The Rand index (alternatively known as Sokal and Michener simple matching coefficient) was selected due to its direct applicability to the binary data representation of the pin configurations. The Rand index provided a base measure that considered both the positive and negative matches; unlike, for instance, the comparative Jaccard index, which only considers positive matches [101]. This was pertinent to the application of the research, as the practical outcome of both positive and negative matches are equally beneficial to the objective; both types of matches ensure that the specific hole in the base plate array can remain as-is for the successive reconfiguration, thus minimising the reconfiguration effort required for the task.

The binary dissimilarity measure implemented in the research was a modification of the Rand index; this is shown in Equation (6.3).

$$D = 1 - \frac{a + d}{a + b^2 + c^2 + d} \quad (6.3)$$

Where:

D = dissimilarity value

a = number of positive matches (1 to 1)

b = number of positive-to-negative mismatches (1 to 0)

c = number of negative-to-positive mismatches (0 to 1)

d = number of negative matches (0 to 0)

The Rand index was modified with exponential weightings used on the mismatches (b and c). The weightings enforce a harsh penalty on any pin that has to be inserted and/or removed, thus favouring reconfigurations where minimal pin manipulation is required. An abundance of d relationships would be typical, as per the specifications presented in Section 3.4 (where pin configurations were limited to 8 – 16 pins out of 64 holes). Thus, the weightings on the mismatches also restrict this influence, such that the datasets generated exhibit greater variance.

The measure used was converted to a dissimilarity measure by subtracting the ratio from 1. This ensured that the value could be employed with clustering algorithms that were designed for distance measures, where closeness is used as the clustering criteria.

The binary dissimilarity measure compares the matrix of one pin configuration to that of another on an elementwise basis. Examples of the elementwise relationships used in Equation 6.3 are displayed in

Figure 6.4, where the large square-shaped part is shown to be reconfigured to the large triangular-shaped part.

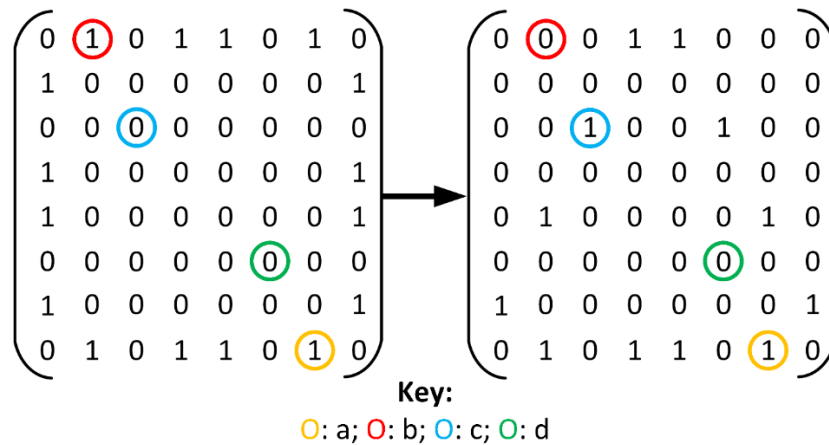


Figure 6.4: Elementwise relationships in practice

The selection of an appropriate similarity/dissimilarity measure is considered subjective, as there is no definitive quantification of validity for one measure over every other [102]. However, Equation (6.3) can be justified by the direct correlation observable for the mechanisms of the dissimilarity measure in relation to the reconfiguration task. The dissimilarity measure can also be justified by the silhouette values generated by the clusters formed in practice; Section 9.3 revealed favourable results for the related tests, where the clusters generated were clear and unambiguous.

Equation (6.3) is applied to compute the dissimilarity values between all pin configurations in the job list. These values are amalgamated into a distance matrix. Table 6.3 shows an example of such a matrix, where pairwise dissimilarities can be retrieved by intersecting the row and column of the desired pair of pin configurations.

Table 6.3: Sample problem non-metric distance matrix

	1	2	3	4	5	6	7	8	9	10	11	12
1	0	0.888889	0.606061	0.888889	0.606061	0.888889	0.825397	0.769231	0.839552	0.780952	0.866013	0.929752
2	0.888889	0	0.825397	0.727273	0.825397	0.727273	0.606061	0.727273	0.810924	0.79646	0.850694	0.666667
3	0.606061	0.825397	0	0.825397	0.727273	0.73913	0.878049	0.825397	0.800885	0.821138	0.839552	0.917431
4	0.888889	0.727273	0.825397	0	0.73913	0.363636	0.825397	0.727273	0.810924	0.832061	0.82	0.808333
5	0.606061	0.825397	0.727273	0.73913	0	0.825397	0.727273	0.606061	0.800885	0.821138	0.908867	0.856164
6	0.888889	0.727273	0.73913	0.363636	0.825397	0	0.825397	0.727273	0.844203	0.832061	0.82	0.867089
7	0.825397	0.606061	0.878049	0.825397	0.727273	0.825397	0	0.210526	0.755208	0.730337	0.804348	0.679487
8	0.769231	0.727273	0.825397	0.727273	0.606061	0.727273	0.210526	0	0.718391	0.752577	0.782407	0.808333
9	0.839552	0.810924	0.800885	0.810924	0.800885	0.844203	0.755208	0.718391	0	0.800885	0.853147	0.760204
10	0.780952	0.79646	0.821138	0.832061	0.821138	0.832061	0.730337	0.752577	0.800885	0	0.837121	0.878788
11	0.866013	0.850694	0.839552	0.82	0.908867	0.82	0.804348	0.782407	0.853147	0.837121	0	0.866013
12	0.929752	0.666667	0.917431	0.808333	0.856164	0.867089	0.679487	0.808333	0.760204	0.878788	0.866013	0

The four properties that constitute a metric distance matrix are listed as follows (where i and j denote the row and column of the matrix, respectively) [103]:

- $x_{ii} = 0$ for all i , i.e. main diagonal entries are all zero;
- $x_{ij} > 0$ if $i \neq j$, i.e. off-diagonal entries are all positive;
- $x_{ij} = x_{ji}$, i.e. the matrix is symmetric; and
- $x_{ij} \leq x_{ik} + x_{kj}$ for all k , i.e. the triangle inequality is satisfied.

The distance matrices generated from the dissimilarity values satisfy these conditions, except for the triangle inequality. Therefore, the values are unsuitable for plotting in real space without modification, since the data was non-Euclidean. The distance matrix generated is, thus, a non-metric distance matrix. However, clustering algorithms predominantly rely on grouping data based on their geometric closeness [102].

Hierarchical clustering was investigated, which is a technique applicable to non-metric data. However, this technique is appropriate for ranking of data rather than grouping, since the separation of groups is not explicitly defined. The algorithm is a greedy algorithm, so backtracking to find better solutions is not possible; this would affect the optimality of the result [102]. The k-means algorithm was suitable for the application, as the data is iteratively grouped into a predefined number of clusters. The algorithm also considers outliers, which are commonly undesirable, but necessary for the purpose of the research undertaken; this is because the assignments have to be made to include all members of the dataset, such that every part is assigned to a fixture and none are ignored. However, the k-means algorithm is only applicable to data expressed in two-dimensional real space [102].

Non-metric MDS was utilised to solve the problem of scaling the non-metric distance matrix to a two-dimensional plane, such that the k-means algorithm could be applied. MDS is an ordination technique where multi-dimensional data is scaled to a lower dimensional plane, while preserving the relative distances of the original data [103]. This was executed in MATLAB® 2016a by minimising Kruskal's normalised stress criterion for Equation (6.4) [104].

$$Str = \sqrt{\frac{\sum_{i,j} (d_{ij} - \tilde{d}_{ij})^2}{\sum_{i,j} d_{ij}^2}} \quad (6.4)$$

Where:

Str = Kruskal's normalised stress, or Stress-1

d_{ij} = raw dissimilarity values for each pair of objects

\tilde{d}_{ij} = scaled distances in the required number of dimensions for each pair of objects

The MDS procedure minimises the stress value, which represents the goodness of the scaled data in comparison to the original data. A stress value of '0' means a perfect scaling of the data. However, scaling from higher-dimensional planes rarely yields a perfect score. Sturrock and Rocha [105] developed an evaluation table from which the final stress value could be compared to assess the relevancy of the scaling outcome. This threshold value depends on the number of dimensions scaled to and from.

The result of the non-metric MDS procedure yields a two-dimensional plane that represents the non-metric distance matrix. An example of this is shown in Figure 6.5, where the distance matrix from Table 6.3 was scaled to two dimensions, with data points representing parts.

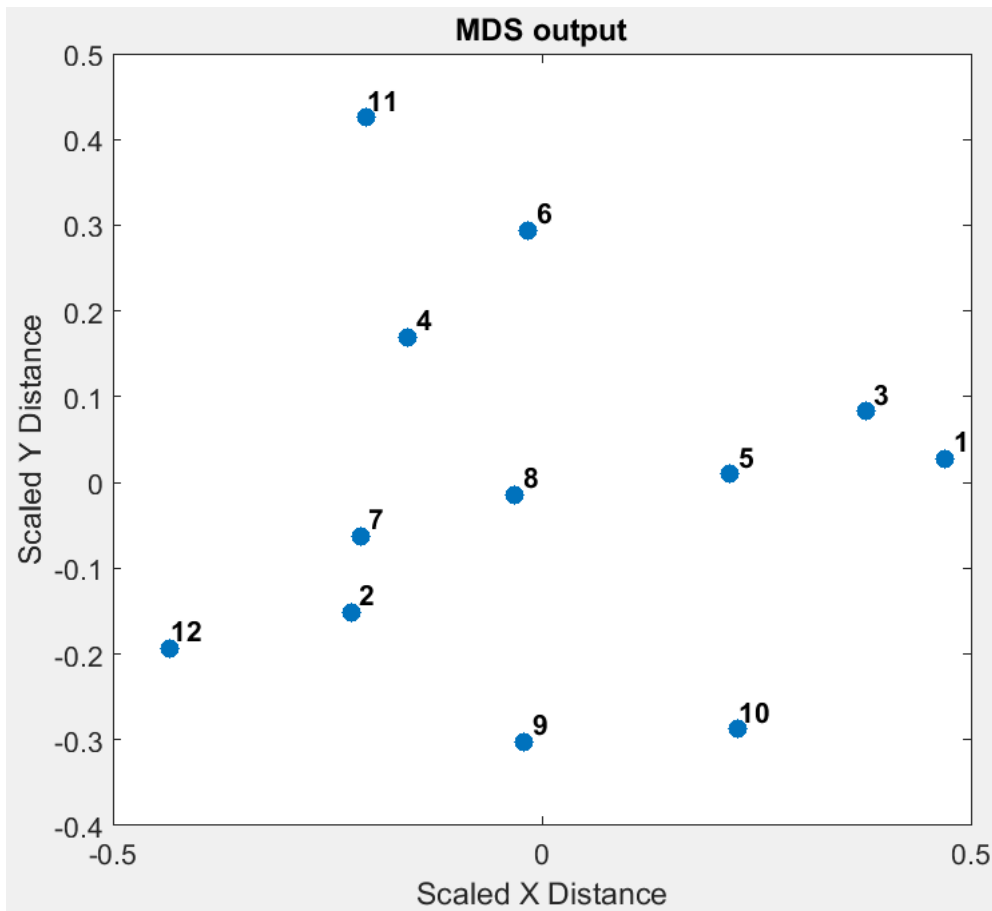


Figure 6.5: Sample problem MDS output

The resultant two-dimensional map in Figure 6.5 can then be used to group the n number of data points to m number of clusters, which are based on the number of parts and fixtures, respectively. This was done in MATLAB® 2016a by using Lloyd’s algorithm for k-means clustering [106]. The k-means procedure is presented by the pseudocode below (Figure 6.6) [102], where m is used instead of k as per the notation displayed in Section 6.5.

k-means Algorithm
<p>Input: n (instance set), m (number of cluster)</p> <p>Output: clusters</p> <ol style="list-style-type: none"> 1. Initialize m cluster centres. 2. while termination condition is not satisfied do 3. Assign instances to the closest cluster centre. 4. Update cluster centres based on the assignment. 5. end while

Figure 6.6: k-means Algorithm

The termination condition in this case is the sum of intracluster distances to the cluster centroid. The distance measure used was ‘squared Euclidean’, which is the shortest (direct) distance to and from the points considered. The cluster centroid is iteratively updated until the total sum of intracluster distances is minimised. The initialisation of these centroids was executed using the k-means++ procedure for centroid initialisation. This technique improves the speed and accuracy of the clustering procedure. The pseudocode for k-means++ is displayed below (Figure 6.7) [107].

k-means++ Algorithm
<ol style="list-style-type: none"> 1. Choose an initial centre c_1 uniformly at random from data points X. 2. For each data point x, compute $D(x)$, the distance between x and the nearest centre that has already been chosen. 3. Choose the next centre c_i, selecting $c_i = x' \in X$ with probability $(D(x')^2) / (\sum_{x \in X} D(x)^2)$. 4. Repeat Step 3 until a total of k centres have been chosen. 5. Proceed as with the standard k-means algorithm.

Figure 6.7: k-means++ Algorithm

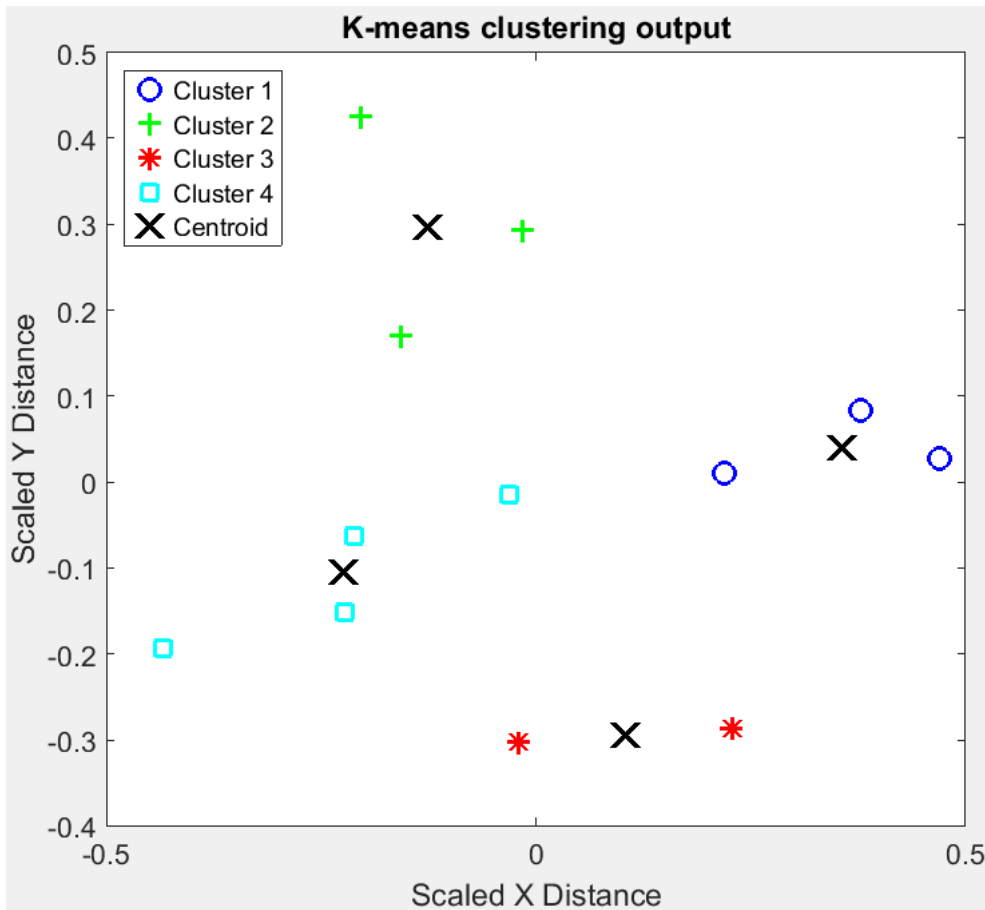


Figure 6.8: Sample problem k-means output. Data clustered to four clusters.

A heuristic was employed to deal with oversized clusters. This problem arises when most data points are highly condensed in an area on the 2D map or when very few (say two) fixtures are available. This would lead to infeasible solutions in Stage III, where a fixture can be used only once every two time periods (as it must be operated on in Cell 1 and Cell 2 thereafter). Therefore, cases where the largest cluster is more than one greater in size than the total sum of every other cluster size, must be amended. The heuristic for identifying and amending this situation is denoted the ‘cluster size fail-safe heuristic; it ensures that there is always another fixture available for use while the most utilised fixture (largest cluster size) is in circulation, thus preventing infeasible solutions of this type in Stage III. The pseudocode for is displayed below (Figure 6.9).

Cluster Size Fail-Safe Algorithm
<pre> 1. Compute sizes of clusters formed from k-means. 2. Subtract total sum of other cluster sizes from the largest cluster size to find deficit <i>def</i>. 3. if <i>def</i> > 1 do 4. Find <i>def</i> object/s furthest away from largest cluster centroid 5. Assign object/s to closest other cluster centroid/s 6. Update cluster data 7. end if </pre>

Figure 6.9: Cluster Size Fail-Safe Algorithm

To conclude Stage I: the pin configurations were represented as binary matrices; these were compared to each other via a binary dissimilarity measure, from which a non-metric distance matrix was generated; this data was scaled to two dimensions so that pin configurations were represented in real space, based on their dissimilarity; this map was then used to group the pin configurations into clusters representative of the fixtures they were to be reconfigured on; this produced the final result, indicating which pin configurations (and thus parts) were to be assigned to which fixture, i.e. the function of Stage I.

6.8 Intracluster Sequencing (Stage II)

The Clustering stage assigns similar parts to the same group. The sequence in which those parts are to be reconfigured on the fixture is determined thereafter by the Intracluster Sequencing stage. The Intracluster Sequencing stage optimally rearranges the reconfiguration order on each fixture.

The MDS plot from which the groups are formed represent the relationships between all members of the dataset; as non-Euclidean measurements, these values are not fully in agreement with each other. There is an inherent accuracy error in the MDS plot by virtue of the stress value being greater than zero. Therefore, the sequencing of parts within that group should refer back to the pairwise dissimilarity values in the non-metric distance matrix, instead of the distances generated from the MDS plot.

Hierarchical clustering is applicable to ranking of data as opposed to grouping, as mentioned in Section 6.7. Therefore, agglomerative hierarchical clustering was used, together with optimal leaf ordering, to optimally sequence the intracluster order.

Agglomerative clustering successively constructs a dendrogram (or tree) by merging the objects (or leaves) into sub-clusters (or subtrees), based on some distance criterion. Single linkage was used, as this resulted in the least dissimilar objects being linked to each other to build the dendrogram [102].

The default order of the final dendrogram does not necessarily represent the optimal order. This could arise from the default order in which the tree was built, where (for example) the first object may be the basis of linkage to the third object, with the second object separating them; or because the agglomeration may have resulted from several leaves being merged to a subtree due to their closeness to a single leaf in that subtree [102]. For n number of leaves, there are 2^{n-1} possible linear orderings of the tree leaves that are consistent with the arrangement of the tree [108]. Bar-Joseph et al. [108] developed an algorithm

to optimally reorder the tree, such that the total cumulative pairwise dissimilarities are minimised. This is akin to finding the shortest path required to traverse every object. The algorithm uses internal node flipping to rearrange the dendrogram and assess if an improvement in the distance traversed is observed.

The algorithm was implemented in MATLAB® 2016a to obtain the optimal leaf order for the dendrogram representing each cluster; this produces the optimal intracluster sequence. The sequence ensures that the most-similar pin configurations in a given cluster are sequenced successively, such that the reconfiguration effort over the fixture’s lifespan for that job is minimised. An example of the implementation of the algorithm is shown in Figure 6.10. Internal node flipping is displayed by the red circle, where the leaf ordering is changed to obtain a different ordering while preserving the tree structure [108].

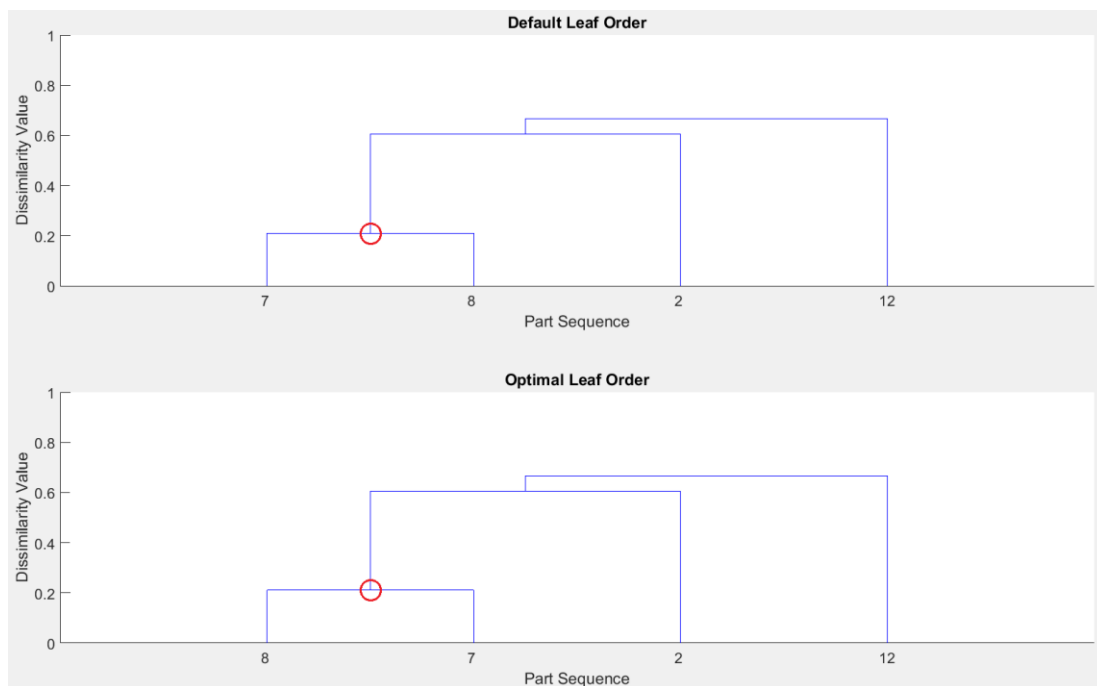


Figure 6.10: Reordering of Fixture 4 in sample problem

Figure 6.10 shows that the default order placed Part 2 after Part 8. However, the algorithm resolved that Part 7 should precede Part 2 since their pairwise dissimilarity value is lower than the original case; while the changeover from Part 7 to Part 8 remains the same since the distance matrix is symmetric. This can be validated with the non-metric distance matrix in Table 6.3.

The procedure is repeated for each cluster, where the data for the parts in that cluster is retrieved from the non-metric distance matrix to form a new distance matrix for that fixture. This condensed distance matrix is then used by the agglomerative hierarchical clustering algorithm, before the optimal leaf ordering algorithm is employed. The resultant sequence generates the reconfiguration order for which the total cumulative pairwise dissimilarity is the minimum for that fixture. This minimises the total reconfiguration effort required across the fixture’s lifespan for that job list, thus minimising the reconfiguration time required.

6.9 Final Scheduling (Stage III)

The preceding two stages yield a matrix in which the fixture-part mappings are arranged as per the intracluster sequences. This is the set I described in Section 6.5, with example shown in Table 6.2. The Final Scheduling stage performs the scheduling function for the two-cell JIT manufacturing system described in Figure 3.5 and Table 6.1. This is achieved by solving a Mixed Integer Linear Programming (MILP) model with a Branch and Bound (B&B) algorithm. The MILP model inputs are the restrictions implied by the I matrix and the predetermined fixture reconfiguration and part processing times. The output is the order in which the fixture-part mappings should be dispatched such that the job list is completed with minimum idle time losses. The following sections present, describe and discuss the MILP model.

6.9.1 MILP Model (Non-Linearised)

The non-linearised MILP model developed for the Final Scheduling stage of the optimisation method was as follows [100]:

$$\text{Min} \sum_i^m \sum_{\hat{i} \neq i}^m \sum_j^{|i|} \sum_{\hat{j}}^{|i|} \sum_k^{n-1} Z_{\hat{i}\hat{j}k\check{k}}, \quad (\forall \check{k} = k + 1) \quad (6.5)$$

Subject to:

$$Z_{\hat{i}\hat{j}k\check{k}} = |(\tau_{ij} - \rho_{\hat{i}\hat{j}}) * Y_{\hat{i}\hat{j}k\check{k}}|, \quad (\forall \check{k} = k + 1) \quad (6.6)$$

$$Y_{\hat{i}\hat{j}k\check{k}} = X_{ijk} * X_{\hat{i}\hat{j}\check{k}}, \quad \left(\begin{array}{l} \forall i, \forall \hat{i} \neq i, \forall j \in i, \forall \hat{j} \in \hat{i}, \\ \forall k < n, \\ \forall \check{k} = k + 1 \end{array} \right) \quad (6.7)$$

$$\sum_i^m \sum_{\hat{i} \neq i}^m \sum_j^{|i|} \sum_{\hat{j}}^{|i|} \sum_k^n Y_{\hat{i}\hat{j}k\check{k}} = n - 1, \quad (\forall \check{k} = k + 1) \quad (6.8)$$

$$X_{ijk} + X_{\hat{i}\hat{j}\check{k}} \leq 1, \quad \left(\begin{array}{l} \forall i, \forall j \in i, \\ \forall \hat{j} \in i < j \in i, \forall k, \forall \check{k} > k \end{array} \right) \quad (6.9)$$

$$\sum_i^m \sum_j^{|i|} X_{ijk} = 1, \quad (\forall k) \quad (6.10)$$

$$\sum_k^n X_{ijk} = 1, \quad (\forall i, j \in i) \quad (6.11)$$

$$X_{ijk} \in \{0,1\}, \quad (\forall i, \forall j \in i, \forall k < n) \quad (6.12)$$

$$Y_{\hat{i}j\check{k}\check{k}} \geq 0, \quad \left(\begin{array}{l} \forall i, \forall \hat{i} \neq i, \forall j \in i, \forall \hat{j} \in \hat{i}, \\ \forall k < n, \\ \forall \check{k} = k + 1 \end{array} \right) \quad (6.13)$$

$$Z_{\hat{i}j\check{k}\check{k}} \geq 0, \quad \left(\begin{array}{l} \forall i, \forall \hat{i} \neq i, \forall j \in i, \forall \hat{j} \in \hat{i}, \\ \forall k < n, \\ \forall \check{k} = k + 1 \end{array} \right) \quad (6.14)$$

6.9.2 MILP Model Description

The Objective Function (6.5) aims to optimally match the part processing time for fixture-part mapping $j \in i$ and fixture reconfiguration time for another fixture-part mapping $\hat{j} \in \hat{i}$ for every time period $\check{k} = k+1$ (which determines the fixture-part mapping $j \in i$ to be scheduled for fixture reconfiguration in time period k), such that the final accumulated operation time differences between them is minimised. This synchronises the fixture reconfiguration and part processing operations, such that the total idle time in the system is minimised for a given job list.

Constraint (6.6) calculates the absolute difference between the part processing time related to fixture-part mapping $j \in i$ in Cell 2 and the fixture reconfiguration time related to fixture-part mapping $\hat{j} \in \hat{i}$ in Cell 1 for time period $\check{k} = k+1$ for every valid $Y_{\hat{i}j\check{k}\check{k}}$. This determines the individual idle times available for the objective function to find the optimal solution.

Constraint (6.7) associates valid pairs of fixture-part mappings, which ensures the successive flow of operations from Cell 1 to Cell 2. This is determined by ensuring that the binary decision variables related to fixture-part mappings $j \in i$ and $\hat{j} \in \hat{i}$ for time periods k and $\check{k} = k+1$, respectively (i.e. X_{ijk} and $X_{\hat{i}\hat{j}\check{k}}$), must both be active (equal to 1) for $Y_{\hat{i}j\check{k}\check{k}} > 0$. Thus, only synchronous time periods (where both cells are occupied) are valid for the idle time calculations (via $Y_{\hat{i}j\check{k}\check{k}}$).

Constraint (6.8) ensures that the number of $Y_{\hat{i}j\check{k}\check{k}} > 0$ corresponds to the number of time periods in which Cell 1 and Cell 2 perform operations synchronously, i.e. one less than the total number of jobs ($n - 1$), since the first time period hosts an operation in Cell 1 only (the first fixture reconfiguration). This ensures that only a valid number of operations are synchronised in the time periods available.

Constraint (6.9) imposes the intracluster order (determined in Stage II) on the final schedule, by ensuring that any two fixture-part mappings from the same fixture i ($j \in i$ and $\hat{j} \in \hat{i}$) must appear in time periods relative to each other that correspond to the intracluster order ($\check{k} > k$).

Constraint (6.10) ensures that there is only one fixture-part mapping $j \in i$ assigned to a time period k , and Constraint (6.11) ensures that a fixture-part mapping $j \in i$ is assigned to a time period k only once.

Bound (6.12) enforces a binary condition for the decision variable X_{ijk} .

Bounds (6.13) and (6.14) enforce a non-negativity condition for the slack variables $Y_{\hat{i}j\check{k}\check{k}}$ and $Z_{\hat{i}j\check{k}\check{k}}$, respectively. This ensures that the linearising constraints (Section 6.9.2.1) for these decision variables perform their required function.

The constraints and bounds are subject to the variable sets for which they are valid. These further constrain the problem search space to ensure feasible solutions for the problem.

6.9.2.1 Linearisation

The non-linearised MILP model presented in Section 6.9.1 was linearised to produce a formulation of the problem with improved computational efficiency. A model with linear constraints only is less computationally expensive to solve than a model with quadratic constraints. Solvers for Linear Programming (LP) models are readily available; the algorithms include Simplex, Dual-Simplex and Primal-Simplex [52]. LP solvers can provide the B&B algorithm with bounded solutions at each node in reduced time, thus improving the overall efficiency of the MILP solver [109].

Constraint (6.6) is non-linear due to the absolute value that is calculated. The absolute value is necessary to ensure that the total idle time calculated in Objective Function (6.5) is an accumulation of individual idle times that resulted from both Cell 2 and Cell 1 (which would be negative otherwise). Constraints (6.6a) and (6.6b) are used instead of (6.6), in-tandem with Bound (6.14), to linearise the absolute value. The two linearising constraints ensure that, should the operation time difference value $(\tau_{ij} - \rho_{ij})$ be negative, the contribution of the result from Constraint (6.6a) on the objective function (via $Z_{ijjk\check{k}}$) would be zero, while the non-negative value from Constraint (6.6b) would contribute to the objective function for that time period instead. Conversely, the constraints ensure that (6.6a) contributes to the objective function instead of (6.6b) should the operation time difference be positive. This linearising technique is only valid if $Z_{ijjk\check{k}}$ is bounded to remain non-negative, as shown in (6.14) [110].

$$-Z_{ijjk\check{k}} + (\tau_{ij} - \rho_{ij}) * Y_{ijjk\check{k}} \leq 0 \quad (\forall \check{k} = k + 1) \quad (6.6a)$$

$$-Z_{ijjk\check{k}} - (\tau_{ij} - \rho_{ij}) * Y_{ijjk\check{k}} \leq 0 \quad (\forall \check{k} = k + 1) \quad (6.6b)$$

Constraint (6.7) is non-linear due to the product of two decision variables, which results in a quadratic constraint. The product of binary decision variables X_{ijk} and $X_{ij\check{k}}$ is necessary to ensure that idle time values are only calculated when both variables are active, and thus synchronous in that time period. Furthermore, this ensures that JIT workflow from Cell 1 to Cell 2 for the two operations is upheld. Constraints (6.7a) to (6.7c) are used instead of (6.7), in-tandem with Bound (6.13), to linearise the non-linearity. The three linearising constraints ensure that the slack variable $Y_{ijjk\check{k}}$ can be equal to 1 only if both decision variables are equal to 1. If only one or none of the decision variables are equal to 1, the slack variable value is equal to 0. This linearising technique is only valid if $Y_{ijjk\check{k}}$ is bounded to remain non-negative, as shown in (6.13) [109].

$$-Y_{ijjk\check{k}} + X_{ijk} + X_{ij\check{k}} \leq 1 \quad \left(\begin{array}{l} \forall i, \forall \hat{i} \neq i, \forall j \in i, \forall \hat{j} \in \hat{i}, \\ \forall k < n, \\ \forall \check{k} = k + 1 \end{array} \right) \quad (6.7a)$$

$$Y_{ijjk\check{k}} - X_{ijk} \leq 0 \quad \left(\begin{array}{l} \forall i, \forall \hat{i} \neq i, \forall j \in i, \forall \hat{j} \in \hat{i}, \\ \forall k < n, \\ \forall \check{k} = k + 1 \end{array} \right) \quad (6.7b)$$

$$Y_{\hat{i}\hat{j}k\check{k}} - X_{\hat{i}\hat{j}\check{k}} \leq 0 \quad \left(\begin{array}{l} \forall i, \forall \hat{i} \neq i, \forall j \in i, \forall \hat{j} \in \hat{i}, \\ \forall k < n, \\ \forall \check{k} = k + 1 \end{array} \right) \quad (6.7c)$$

6.9.2.2 Decision Variable Indices

Table 6.4 shows an updated version of the workflow table from Table 6.1, with the decision variable indices used instead of alphabets. The model examines the events in a synchronous time period (say $k = 2$), and considers this scenario in order to schedule the reconfiguration operation in the previous time period ($k = 1$, in said case); which, in itself, schedules the reconfiguration operation for the current time period ($k = 2$, in said case). Therefore, the scheduling of every time period is reliant on the occurrences in the next time period, which makes finding the optimal solution computationally difficult.

Table 6.4: Workflow table with decision variable indices

Time Period	Cell 1		Cell 2	
	Fixture Reconfiguration ($\rho_{\hat{i}\hat{j}}$)		Part Process with Fixture (τ_{ij})	
1	$X_{\hat{i}\hat{j}\check{k}}$			
2	$X_{\hat{i}\hat{j}\check{k}}$		X_{ijk}	
3	$X_{\hat{i}\hat{j}\check{k}}$		X_{ijk}	
4	$X_{\hat{i}\hat{j}\check{k}}$		X_{ijk}	
			X_{ijk}	

Table 6.5 shows a modified version of the workflow table from Table 6.4, with example indices used for illustration. The rightmost column displays how the $Z_{\hat{i}\hat{j}k\check{k}}$ slack variable is produced from the valid decision variables for a time period. This slack variable holds the individual idle time values for synchronous time periods in the final solution (as per Constraint (6.6)). Table 6.5 uses colour-coding on the indices of the slack variable to indicate how the final schedule was interpreted from the resultant $Z_{\hat{i}\hat{j}k\check{k}}$ outputs. The variables sets defined for Equations (6.5) to (6.14) are crucial for ensuring that these indices remain within feasible boundaries.

Table 6.5: Workflow table with example indices

Time Period	Cell 1	Cell 2	
	Fixture Reconfiguration ($\rho_{\hat{i}\hat{j}}$)	Part Process with Fixture (τ_{ij})	
1	X_{111}		
2	X_{212}	X_{111}	$Z_{\hat{i}\hat{j}k\check{k}} = Z_{121112}$
3	X_{123}	X_{212}	$Z_{\hat{i}\hat{j}k\check{k}} = Z_{211223}$
4	X_{224}	X_{123}	$Z_{\hat{i}\hat{j}k\check{k}} = Z_{122234}$
		X_{224}	

6.9.3 Solution Technique

The MATLAB® 2016a MILP solver (*intlinprog*) was used to formulate and solve the MILP problem. The solver uses the branch and bound solution technique to solve the problem, as described in Section 2.7.2. The pseudocode for the B&B method is presented below (Figure 6.11) [111].

Branch and Bound Algorithm	
1.	begin
2.	<i>activeset</i> := {0};
3.	<i>bestval</i> := {0};
4.	<i>currentbest</i> := {0};
5.	while <i>activeset</i> is not empty do
6.	choose a branching node, node $k \in \textit{activeset}$;
7.	remove node k from <i>activeset</i> ;
8.	generate the offspring of node k , offspring i , $i = 1, \dots, n_k$, and corresponding optimistic bounds ob_i ;
9.	for $i = 1$ to n_k do
10.	if ob_i worse than <i>bestval</i> then terminate offspring i ;
11.	else if child is a complete solution then
12.	<i>bestval</i> := ob_i , <i>currentbest</i> := offspring i ;
13.	else add offspring i to <i>activeset</i>
14.	end for
15.	end while
16.	end

Figure 6.11: Branch and Bound Algorithm

The MILP solver was adjusted to impose the selected options. The options selected for the research implementation are shown in Table 6.6.

Table 6.6: MILP options used

Option Name	Option Selected
Branch Rule	Most Fractional
Constraint Tolerance	1×10^{-3}
Cut Generation	None
Heuristics	RSS Hybrid
Integer Pre-Process	None
Integer Tolerance	1×10^{-3}
LP Optimality Tolerance	1×10^{-6}
Node Selection	Min Objective Function
Objective Improvement Threshold	1×10^{-3}
Relative Gap Tolerance	1×10^{-3}
Root LP Algorithm	Primal-Simplex

The Branch Rule selected the components whose fractional part was closest to a value of half; rough testing revealed that this option reliably converged to the solution by exploring fewer nodes than the other available options. The Constraint Tolerance value was relaxed due to the integer values used for the problems tested. Cut Generation was eliminated due to the complexity of problem, as its absence availed a greater solution search space; this was employed due to the highly constrained formulation of the problem. Furthermore, rough testing revealed that Cut Generation produced only marginal improvements in the initial Lower Bound (LB), at the expense of increased solutions times for the procedure. The Heuristic was used to find the LP relaxation solution for the root LB; rough testing revealed that the hybrid (a combination of local branching and the technique developed by Danna et al. [112]) yielded the fastest solutions. Integer Pre-Processing was eliminated for the same reasons as Cut Generation. The LP Optimality Tolerance was adjusted for improved precision. The Node Selection favoured nodes with the minimum objective function value; this was chosen for the same reason as the Branch Rule. The Objective Improvement Threshold and Relative Gap Tolerance were adjusted for the same reason as the Constraint Tolerance. The Primal-Simplex algorithm was selected; rough testing yielded solutions with reduced node exploration and solution time over the Dual-Simplex algorithm.

6.10 Stage III Heuristic

Section 9.5 demonstrates the computational expense of solving the MILP model with the B&B algorithm. The B&B algorithm is an exact method that extensively searches the state space for the optimal solution. The computational expense of the B&B algorithm grows in polynomial time with the state space size [113]. Thus, a heuristic was developed to solve the Final Scheduling stage faster than what was exhibited in Section 9.5. The Stage III Heuristic (S3H) employed a greedy algorithm approach on the I matrix obtained after Stage II; this produced near-optimal solutions.

Figure 6.12 shows the algorithm flowchart for the Stage III Heuristic. The I matrix from Stage II displayed fixtures as rows and parts as elements in those rows; this matrix had to be transposed for the heuristic, so that matrix element numbers could be utilised in MATLAB® 2016a (since the software counts elements in a matrix by moving down the columns). The operation time matrices were represented by R and T instead of ρ and τ , respectively, as this was a more usable notation for naming the matrices.

Figure 6.12 explains the matrix manipulations executed by the Stage III Heuristic. The S3H was designed to find the best pair of fixture-part mappings for each time period by assessing the first available candidates from each fixture, i.e. a greedy algorithm approach was used. The objective is achieved by randomly selecting a ‘head’, or first element of each column (which corresponds to the first part sequenced on that fixture). The selected head was denoted the ‘pivot’; while the unused heads were denoted the ‘candidates’. The initial pivot represents the first fixture-part mapping to be reconfigured in the first time period, which means that its part processing operation occurs in the second time period. The S3H compares the part processing time of the pivot to the fixture reconfiguration times of each candidate. The candidate that produces the minimum absolute time difference is selected as the fixture-part mapping to be reconfigured in that time period. The previous pivot is erased from future selection, while the succeeding element becomes the new candidate for that fixture (ensuring that the intracluster sequence is maintained). The best candidate that was selected becomes the new pivot, whose part processing time is then compared to the reconfiguration times of the updated candidates. The procedure continues until the elements of the I matrix are exhausted, as no further valid candidates are left (due to erased pivots). The individual idle times at each step are accumulated to produce the total idle time for the resultant schedule. The procedure described is illustrated in Figure 6.13 where the pivots are shown in red and the candidates are shown in blue. Figure 6.13 is read from top left to bottom right.

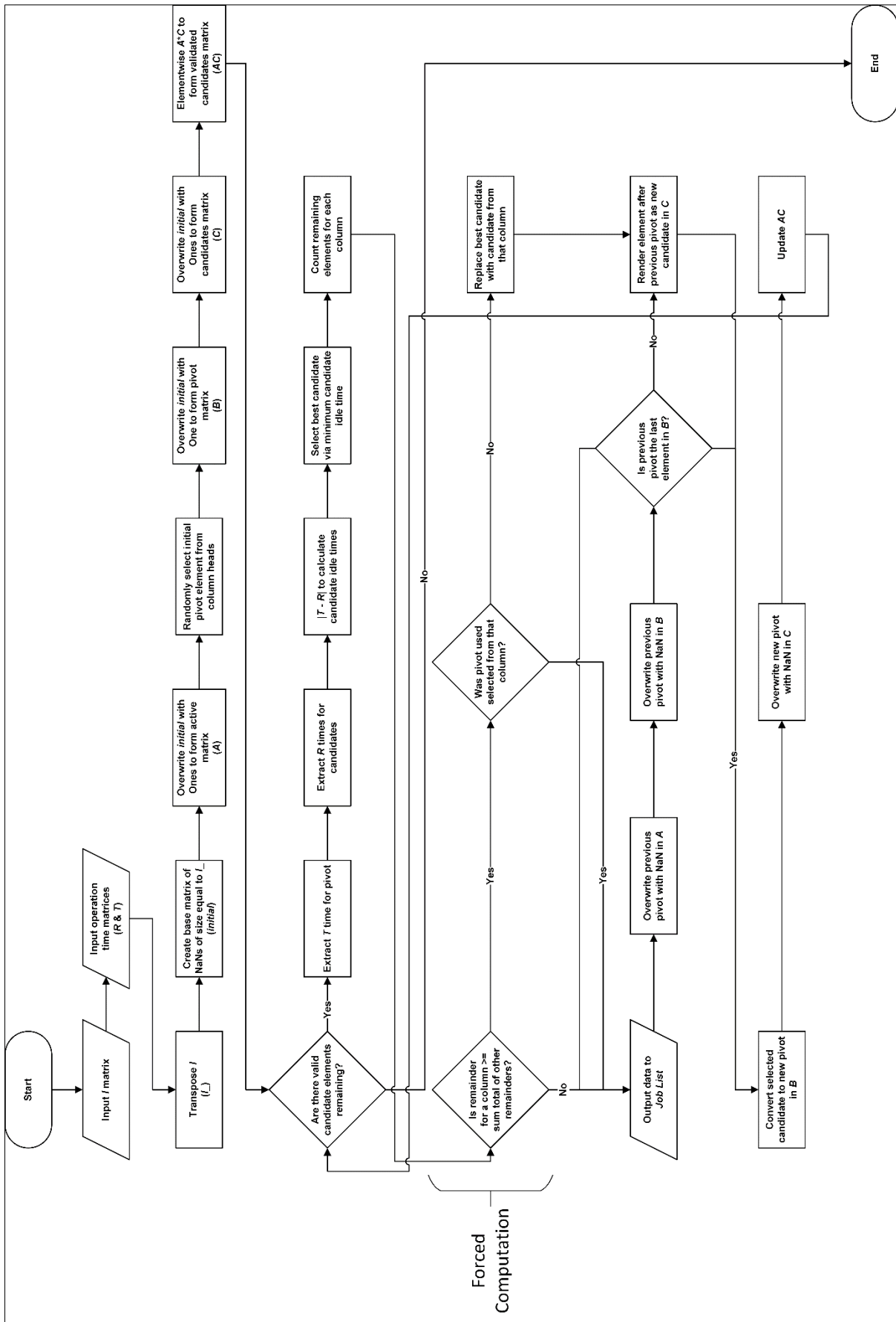


Figure 6.12: Stage III Heuristic flowchart

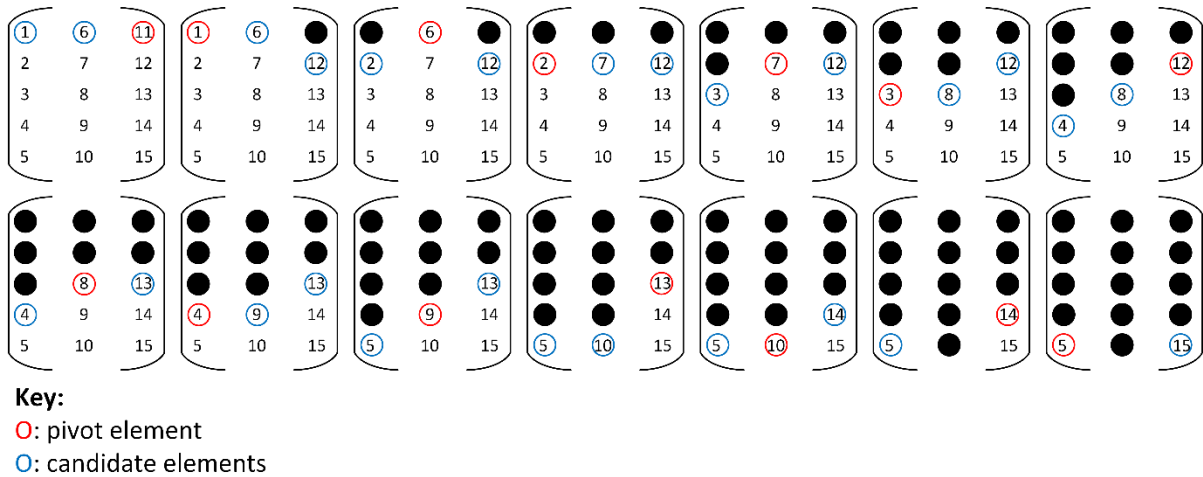


Figure 6.13: Example of Stage III Heuristic actions on the transposed I matrix

A ‘forced computation’ fail-safe heuristic was implemented within the S3H to deal with lag columns (as shown in Figure 6.12). Lag columns arise when the candidate for said column is repeatedly ignored for selection by the algorithm, due to the idle time calculation favouring other candidates for several steps. The lag column behaviour becomes critical when the number of unused elements in that column is greater than the summation of remaining elements from all other columns; this is shown by the scenario in Figure 6.14. If the solution procedure reaches the lag column critical point, all other columns would be exhausted and the lag column pivot would have no candidates to compare itself to. The two-cell JIT production system relies on synchronous operations, which require a fixture in each cell for every time period (except for the first and last). Having one fixture left to complete the job list would result in that fixture being circulated for both cells (only present in one cell per time period), which would significantly reduce utilisation and increase idle time; thus, it is an unacceptable scenario.

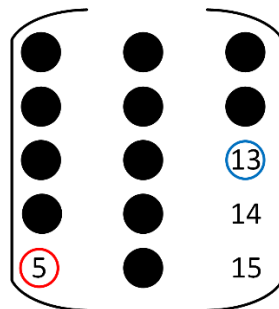


Figure 6.14: Critical lag column (Column 3). Feasible solution unobtainable.

The forced computation fail-safe heuristic checks for the lag column critical point and overwrites the best candidate with the lag column candidate instead. The forced computation increases the total idle time in comparison with the optimal solution, but is necessary to ensure a complete schedule.

The heuristic produced superior solution time performance results to the MILP model with B&B solution, as shown in Section 9.6. The S3H code was appended with a routine that wrote the solution in the appropriate format for the AnyLogic® simulation (see Appendix B.3), so that large-sized schedules could be tested.

Thus, the S3H can solve the Final Scheduling stage faster, but less accurately, than the MILP model. The individual idle times are minimised at each step, but this solution technique does not guarantee the lowest total idle time for the job list. The greedy approach eliminates the possibility of backtracking to find better solutions than those already applied. The lack of backtracking also necessitates the need for a forced computation to be made for certain conditions, to ensure that a feasible schedule can be obtained. The S3H produces good, but sub-optimal, solutions as a result of the greedy approach used; however, the solution times achieved are significantly lower.

6.11 Summary

This chapter discussed the optimal scheduling method developed for the fixture manufacturing cell. The method was separated into three stages which: optimally assigned parts to fixtures; optimally sequenced those parts on their assigned fixtures; and optimally scheduled the fixture and part operations such that total idle time was minimised. These stages required: binary data manipulations; multidimensional scaling; k-means clustering; agglomerative hierarchical clustering with optimal leaf ordering; and a mixed integer linear programming model solved through the branch and bound method. An alternative heuristic (S3H) was developed with a greedy algorithm approach to solve the Final Scheduling problem in reduced time, for near-optimal solutions. This work represents a major contribution of the research, considering the findings in Section 2.8.2 of the gap in research.

7 Agent-Based Simulation

7.1 Introduction

This chapter presents the agent-based simulation for the fixture manufacturing cell (as per the fifth objective in Section 1.4). The simulation was created to model the behaviour of the FxMC in light of the limitations of the proof-of-concept discussed in Section 4.6. The simulation was also required for testing large-sized problems, which would otherwise be practically difficult to conduct. The simulation was developed in AnyLogic® 8.1.0, which uses the Java® programming language to execute commands.

7.2 Simulation FxMC Layout

The FxMC layout described in Figure 3.4 and implemented in Chapter 4 had to be modified for its representation in the simulation. Space Markup shapes were used to construct the cell layout in the software; these included paths, nodes and pallet racks; attractors were also used to place and orientate the agents correctly. The simulation FxMC layout is shown in Figure 7.1, with labels corresponding to the node names, with clarification in brackets to explain the associations with Figure 3.4 and the Lab FxMC.

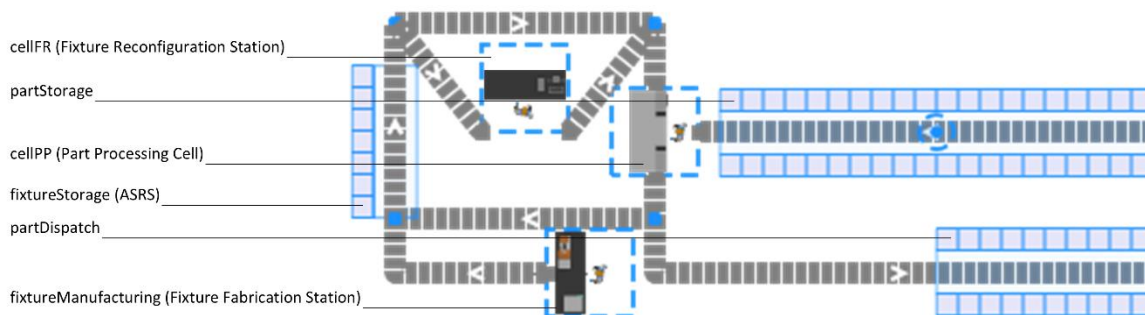


Figure 7.1: Simulation FxMC layout, with Space Markup labels

Figure 7.1 describes the FxMC layout that could best represent Figure 3.4 in the software used. The Fixture Fabrication Station had to be separated from the Fixture Reconfiguration Station by a conveyor used for the return of fixtures. This was due to complications that were experienced when trying to intersect paths in the software. However, as the Fixture Fabrication Station was used only for the initialisation of the simulation run, the modification had minimal influence on the overall behaviour of the cell. The Space Markup for the Fixture Reconfiguration Station was denoted ‘cellFR’ as this is the main function of the fixture manufacturing cell (Cell 1, as per Figure 3.5). Apart from the pathway modification, Figure 7.1 resembles a similar layout to that in Figure 3.4, with ASRS of the same capacity as the one used, and a CNC machine used to represent the Part Processing Cell (denoted ‘cellPP’); there are also storage racks used to represent the Part Storage and Part Dispatch aspects of the production

system. The arrows on the conveyors show the workflow direction along those paths; the arrows follow the unidirectional workflow designated for the cell in Section 3.5.

The simulation software describes the locations where operations are performed as nodes; these would correspond to the Fixture Fabrication Station, Fixture Reconfiguration Station, and Part Processing Cell. The conveyors are referred to as paths, on which agents are transported. The agents for the simulation would be the fixtures and the parts. The storage racks are not strictly nodes, but rather a location along a path where agents are put aside for a specified period of time.

Figure 7.2 shows a screenshot from the animation of the agent-based simulation GUI. The image shows an isometric view of the cell layout, with fixtures shown as red pyramids and parts shown as green spheres. Part Storage is partially shown on the right, with the conveyor leading to the operator for the part processing cell. Part Dispatch is further down the conveyor leading out of the image. The out-of-view components were large storage racks for the holding of finished and unfinished parts, respectively.

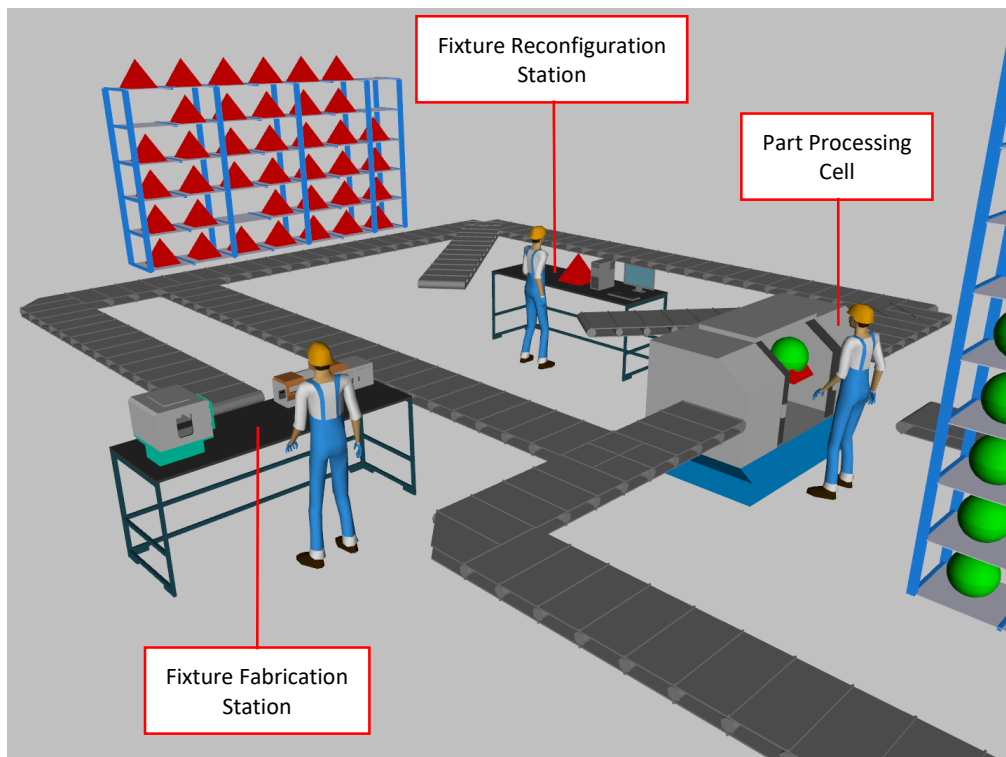


Figure 7.2: Animation screenshot of cell layout

Figure 7.3 shows the fixture fabrication station in closer detail. A separate operator was used due to the separation of this station from the Fixture Reconfiguration Station. The CNC router is shown to the right of the operator, and a 3D printer was also included to the left of the operator. The figure shows the operator working on the fabrication of a fixture, which is held at this station for a certain period of time. See Appendix C.1 for the Standard Operating Procedure (SOP) to be followed for this station.

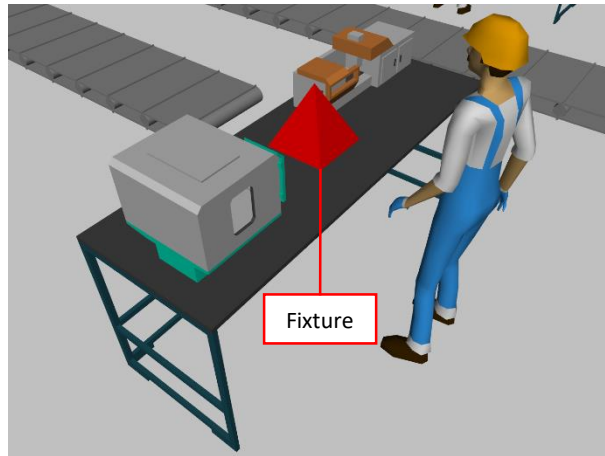


Figure 7.3: Fixture Fabrication Station

Figure 7.4 shows the Fixture Reconfiguration Station in closer detail. A PC is shown on the worktable, as was the case for the Lab FxMC in Chapter 4. The space to the left of the operator is shown to be used for the reconfiguration of a fixture, where it is held for a period of time that corresponds to the fixture reconfiguration time for that pin configuration. See Appendix C.2 for the SOP to be followed for this station.

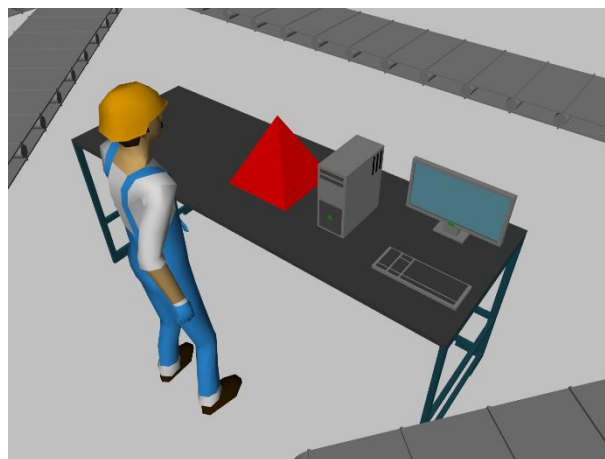


Figure 7.4: Fixture Reconfiguration Station

Figure 7.5 shows the Part Processing Cell in closer detail. The cell is represented by a single CNC machine. The fixture is received along the conveyor and stops inside the machine. A part would be waiting for the fixture at this point. Upon arrival of both the fixture and part at this location, the agents are held for a time corresponding to the part processing time for that fixture-part mapping. This station is not a component of the fixture manufacturing cell, but is required for the representation of Cell 2 in the scheduling method. See Appendix C.3 for the SOP to be followed for this cell.

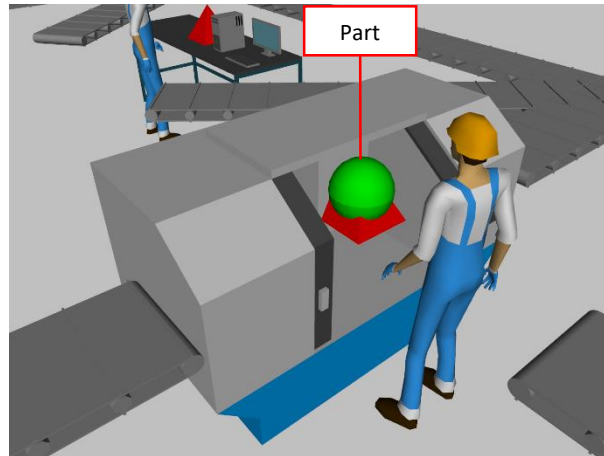


Figure 7.5: Part Processing Cell

7.3 Agents and Resources

The simulation relies on the use of agents, which move along paths to nodes, and seize available resources in the system when required. The agents implemented in the simulation were:

- Fixtures, and
- Parts

The agents were created from databases; these were Excel® spreadsheets generated by the MATLAB® Stage III Heuristic code and linked to the AnyLogic® simulation (see Appendix B.3). The spreadsheets used to create the agents were:

- fixture_generation.xls, and
- part_generation.xls.

The fixtures were generated with four parameters incorporated into them, which are used to identify specific agents for particular tasks. The fixture parameters were:

- 1) Fixture Name;
- 2) Intracluster Sequence Number;
- 3) Storage Column; and
- 4) Storage Row.

Intracluster Sequence Number is necessary for the correct parameters to be read from the main database for a fixture that has been recirculated (since the fixture name alone would cause ambiguity).

The parts were generated with only one parameter:

- 1) Part Name

The parts were stored in numerical order, and circulated only once in the system. As such, no further uniqueness was required for correct identification.

The parameters were used to identify the agents in the main database spreadsheet:

- main_data.xls

The data read from this spreadsheet comprised of:

- scheduling order;
- fixture to be reconfigured;
- intracluster sequence number for the configuration of that fixture;
- fixture reconfiguration time required;
- part to be processed; and
- part processing time required.

The information in the spreadsheets were generated from the results of the Stage III Heuristic. The MATLAB® code for the S3H contained an appended routine for arranging the results in an appropriate manner for the AnyLogic® simulation databases, and then printing these results to the Excel® spreadsheets (as seen in Appendix B.3). The simulation software updates the databases from the spreadsheets before a simulation is run.

The resources available in the simulation relate to the operators of the fixture manufacturing cell and part processing cell, respectively. The resources were:

- FR, for Fixture Reconfiguration Station; and
- PP, for Part Processing Cell.

As such, the terms ‘cellFR’ and ‘cellPP’ are to be used interchangeably with fixture manufacturing cell and part processing cell, respectively, in reference to the simulation hereof. The resources were seized by the relevant agents when a fixture was to be reconfigured (cellFR) or a part was to be processed on a fixture (cell PP). An agent flow quantity of 1 was used to ensure the unit workflow of the production system (Section 2.6.2 and 6.6) was upheld in the simulation. Therefore, a resource cannot be seized by an agent until the agent it is currently seized by has released it.

7.4 Process Modelling Flowchart

A process-centric model of the production system was created in the software by constructing a flowchart of the processes that used the agents and resources. The flowchart is shown in Figure 7.6.

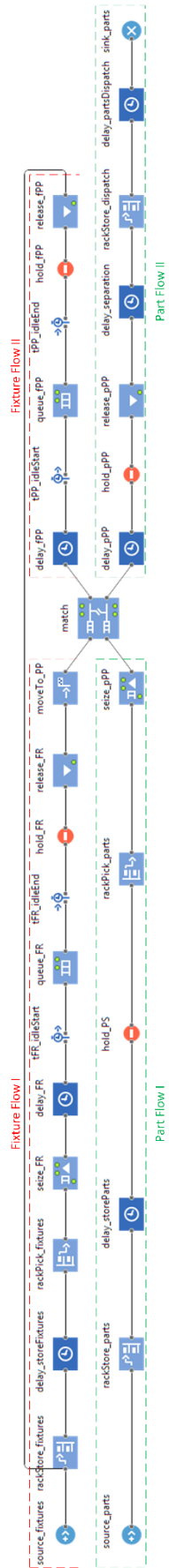


Figure 7.6: Simulation Process Flowchart

The flowchart was divided into five segments, identifiable from Figure 7.6:

- 1) Fixture Flow I
- 2) Part Flow II
- 3) Match
- 4) Fixture Flow I
- 5) Part Flow II

The flowchart was the basis upon which the model derived its behaviour. The processes model the activities of the agents and resources in the system. The following sections discuss the components of Figure 7.6. The basic functions of the flowchart blocks will be described, together with the additional Java® commands that were required.

7.4.1 Fixture Flow I

Fixture Flow I deals with the creation, storage and reconfiguration of fixtures. The segment includes the processes required before the fixture is used to secure the unfinished part it is assigned to. Table 7.1 describes the functions of the blocks from this segment.

Table 7.1: Flowchart block descriptions for Fixture Flow I













Flowchart Block	Basic Description	Additional Java Commands
 source_fixtures	Creates the agents 'Fixtures'	Fixtures created from <i>fixture_generation</i> spreadsheet. Fixtures assigned parameters of name, intracluster sequence number, storage column and storage row.
 rackStore_fixtures	Stores fixtures in <i>fixtureStorage</i> pallet rack.	Stores fixtures in the shelf corresponding to its <i>fixture_generation</i> spreadsheet data. Updates intracluster sequence number upon entry, so that recycled fixtures can be identified in the <i>main_data</i> spreadsheet. Unblocks <i>hold_FR</i> block to clear pathway for fixture in <i>FR</i> .
 delay_storeFixtures	Fixtures wait in pallet rack until retrieved.	Retrieves first required fixture as per <i>main_data</i> spreadsheet when number of fixtures in storage matches the number of fixtures created by <i>source_fixtures</i> .
 rackPick_fixtures	Retrieves fixture from pallet rack and transports it to <i>cellFR</i> .	None.
 seize_FR	Seizes the <i>FR</i> resource for the fixture.	Blocks <i>hold_FR</i> upon entry to prevent exit from <i>FR</i> until instructed.
 delay_FR	Fixtures wait in <i>FR</i> for the Fixture Reconfiguration Time duration. This simulated the Fixture Reconfiguration operation.	Retrieves delay time from <i>main_data</i> spreadsheet, as per corresponding identification.
 tFR_idleStart	Begins a timer when fixture traverses this block.	None.
 queue_FR	Retains fixture in <i>cellFR</i> if previous fixture has not yet returned to storage from <i>cellPP</i> , i.e. holds fixture while idle in <i>cellFR</i> .	Unblocks <i>hold_PS</i> upon entry, so that the part for this fixture is released. This is required when <i>cellPP</i> was idle and waiting for the completion of <i>delay_FR</i> .







Table 7.1: Flowchart block descriptions for Fixture Flow I (continued...)

 tFR_idleEnd	Ends timer that began previously. Records the time fixture spent in <i>queue_FR</i> , i.e. the idle time for <i>FR</i> .	None.
 hold_FR	Prevents fixture from releasing resource <i>FR</i> until the block is unblocked elsewhere.	None.
 release_FR	Releases resource <i>FR</i> to avail it for the next fixture.	Retrieves next fixture as per <i>main_data</i> spreadsheet information, provided there are still parts to be processed. Does so by ending <i>delay_storeFixtures</i> for the specified fixture only.
 moveTo_PP	Moves the fixture from <i>cellFR</i> to <i>cellPP</i> .	Blocks <i>hold_fPP</i> and <i>hold_pPP</i> upon exit, so that the fixture and part that are now in <i>cellPP</i> can only leave when instructed.

7.4.2 Part Flow I

Part Flow I deals with the part creation, storage and retrieval. This segment is responsible for the parts until they are secured to their respective fixtures. Table 7.2 describes the functions of the blocks from this segment.


Table 7.2: Flowchart block descriptions for Fixture Flow II

Flowchart Block	Basic Description	Additional Java Commands
 source_parts	Creates the agents 'Parts'	Parts created from <i>part_generation</i> spreadsheet. Parts assigned parameter of name only.
 rackStore_parts	Stores parts in <i>partStorage</i> pallet rack.	Trivial delay employed to ensure parts are stored in the order they are created, so that storage location can be obtained more efficiently by using the part name only.
 delay_storeParts	Parts wait in pallet rack until retrieved.	Stops delay for the first part, as per <i>main_data</i> spreadsheet, when number of parts in storage matches the number of fixtures created by <i>source_parts</i> .
 hold_PS	Prevents parts from being retrieved by <i>rackPick_parts</i> until unblocked elsewhere. This ensures the part only leaves the pallet rack when required.	None.
 rackPick_parts	Retrieves part from storage rack and transports it to <i>cellPP</i> .	Blocks <i>hold_PS</i> upon entry to prevent premature release of subsequent parts.
 seize_pPP	Seizes resource <i>PP</i> for the part.	None.

7.4.3 Match

Match is the intersection point for the fixtures and parts, i.e. where the part is secured to its fixture. This is made up one block, as shown in Table 7.3.





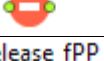
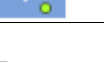
Table 7.3: Flowchart block descriptions for Match

Flowchart Block	Basic Description	Additional Java Commands
<p>match</p> 	Prevents agents from moving further in the flowchart until one of each type have entered the block. Ensures synchronisation of the fixture and part in <i>cellPP</i> .	None.

7.4.4 Fixture Flow II

Fixture Flow II deals with the synchronous delay of the fixture with its part in *cellPP*, before returning the fixture back to *rackStorage_fixtures* for fixture recirculation, as shown in Figure 7.6. Table 7.4 describes the functions of the blocks from this segment.

Table 7.4: Flowchart block descriptions for Fixture Flow II

Flowchart Block	Basic Description	Additional Java Commands
<p>delay_fPP</p> 	Fixture waits in <i>cellPP</i> for the duration of the Part Processing Time it corresponds with. This simulates the Part Process with Fixture operation.	Retrieves delay time from <i>main_data</i> spreadsheet, as per corresponding identification.
<p>tPP_idleStart</p> 	Begins a timer when fixture traverses this block.	None.
<p>queue_fPP</p> 	Retains fixture in <i>cellPP</i> if fixture in <i>cellFR</i> has not yet completed its operation delay, i.e. holds fixture when idle in <i>cellPP</i> .	None.
<p>tPP_idleEnd</p> 	Ends timer that began previously. Records the time fixture spent in <i>queue_fPP</i> , i.e. the idle time for <i>PP</i> .	None.
<p>hold_fPP</p> 	Prevents fixture from releasing <i>PP</i> until unblocked elsewhere.	None.
<p>release_fPP</p> 	Releases resource <i>PP</i> to avail it for the next fixture.	None.

7.4.5 Part Flow II

Part Flow II deals with the synchronous delay of parts with fixtures in *cellPP*, before dispatching the finished parts to Part Dispatch. Part Flow II works in conjunction with Fixture Flow II, as the main tasks correspond to the simultaneous activities of the fixture and part in the part processing cell. Table 7.5 describes the functions of the blocks from this segment.

Table 7.5: Flowchart block descriptions for Part Flow II








Flowchart Block	Basic Description	Additional Java Commands
<p>delay_pPP</p> 	Part waits in <i>cellPP</i> for the duration of the Part Processing Time it corresponds with. This simulates the Part Process with Fixture operation.	Retrieves delay time from <i>main_data</i> spreadsheet, as per corresponding identification. Unblocks <i>hold_pPP</i> and <i>hold_pPP</i> if there are no more parts to process, since activities in Fixture Flow I can no longer unblock these flowchart blocks at that point in the simulation.

Table 7.5: Flowchart block descriptions for Part Flow II (continued...)

<p>hold_pPP</p> 	Prevents part from releasing <i>PP</i> until unblocked elsewhere.	None.
<p>release_pPP</p> 	Releases the <i>PP</i> resource to avail it for the next part.	Retrieves next part as per <i>main_data</i> spreadsheet information, provided there are still parts to be processed. Does so by ending <i>delay_storeParts</i> for that fixture only. Unblocks <i>hold_pPP</i> upon exit so that its fixture can be released at the same time.
<p>delay_separation</p> 	A simple delay to show separation of the part from its fixture on the conveyor flow. The fixture leaves immediately as it is required in <i>rackSystem_fixtures</i> to allow the cell workflow to continue, so the part is delayed.	None.
<p>rackStore_dispatch</p> 	Stores parts in <i>partDispatch</i> pallet rack.	None.
<p>delay_partsDispatch</p> 	Finished parts wait in storage rack for dispatch to customer indefinitely.	Ends simulation upon entry of the final part.
<p>sink_parts</p> 	Used to destroy the Parts agents to remove them from the system when dispatched to customer. Not used in simulations, as parts are held in dispatch indefinitely.	None.

7.5 Simulation Interface

Figure 7.7 shows the GUI that was created for the simulation. The window comprises the following components (numbering corresponds to Figure 7.7):

- 1) Space markup FxMC layout (as described in Figure 7.1), with agent flow animation in 2D;
- 2) Resources, with real-time utilisation data;
- 3) Process flowchart (as shown in Figure 7.6), with current locations of agents in the flowchart, and agent in/out statistics.
- 4) Two horizontal bar graphs showing the number of parts in *partStorage* (unfinished parts) on the left graph, and number of parts in *partDispatch* (finished parts) on the right graph. This data provides information on the current progress of the simulation run.
- 5) FR Utilisation graph. Time plot linked to the utilisation of *delay_FR*, i.e. provides the real utilisation data of *cellFR* (excludes idle time spent in *queue_FR*).
- 6) FR Idle Time graph. Time plot linked to the individual idle times generated by *tFR_idleEnd* for each agent. Can be used to analyse statistics on the real idle time behaviour of the fixtures in the simulation.
- 7) PP Utilisation graph. Time plot linked to the utilisation of *delay_fPP*, i.e. provides the real utilisation data of *cellPP* (excludes idle time spent in *queue_fPP*).

- 8) PP Idle Time graph. Time plot linked to the individual idle times generated by *tPP_idleEnd* for each agent. Can be used to analyse statistics on the real idle time behaviour of the parts in the simulation.
- 9) 3D animation of the simulation. Provides an isometric view of the agent flow during the simulation run.

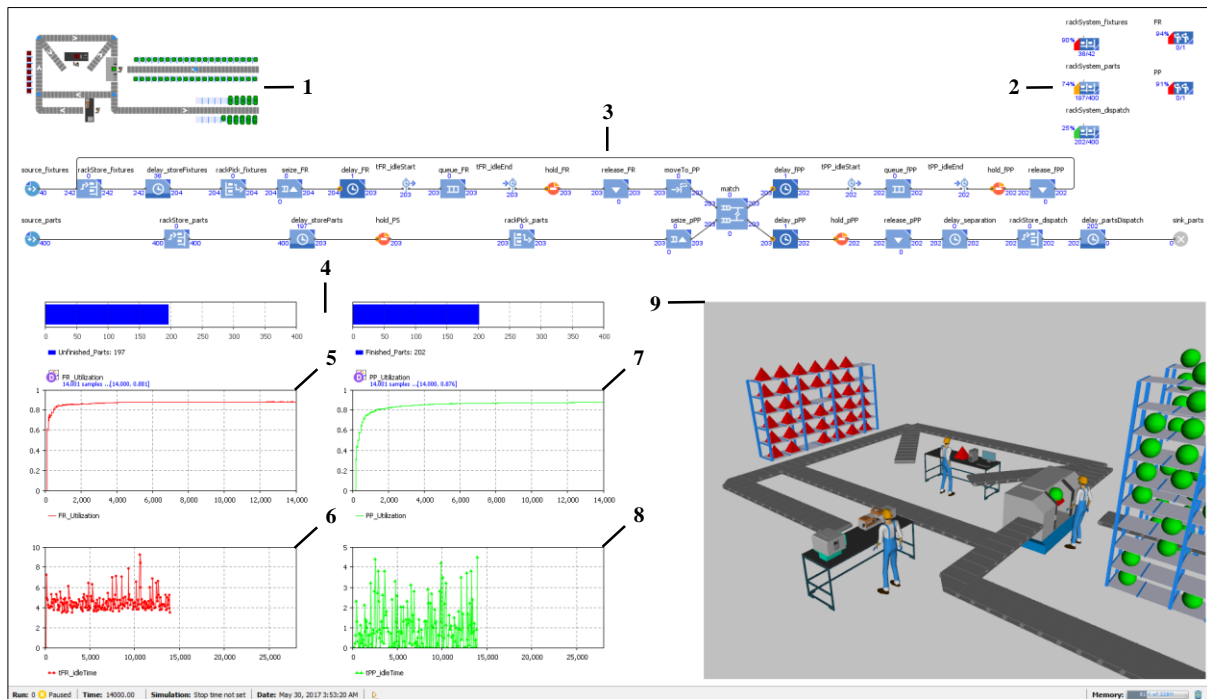


Figure 7.7: Simulation GUI

The utilisation graphs were generated as explained due to the design of the process flowchart; the agents only release their respective resources when they are about to exit the location of that resource, as described in Table 7.1 and Table 7.5. Thus, the idle times spent in the respective queues before release are included in the real-time utilisation statistics of the resources (shown at the top right of Figure 7.7). The graphs, then, provide the utilisation data on the actual time spent by those resources on the operation delays, which is the required statistic.

7.6 Summary

An agent-based simulation was used to model the fixture manufacturing cell. The purpose of the simulation was to overcome the shortcomings of the Lab FxMC (Chapter 4), and to provide an instrument for large-scale testing. The software used for implementation was AnyLogic®. A process flowchart was constructed, together with supplementary Java® commands, to model the expected cell behaviour. A GUI was created from which the simulation run could be monitored.

8 Sample Problem

8.1 Introduction

This chapter presents a sample problem that was solved using the scheduling method described in Chapter 6, before the simulation described in Chapter 7 was used to simulate the schedules generated.

8.2 The Problem

Rough testing revealed that the MILP model could not reliably solve problems with more than 12 fixtures, due to the computational expense of the branch and bound algorithm; as such, a problem of this maximum size was chosen to be solved. The pin configurations corresponding to the parts to be processed are displayed in Figure 8.1. These would relate the customer orders, derived from the CAD/CAM software mentioned in Section 5.4.

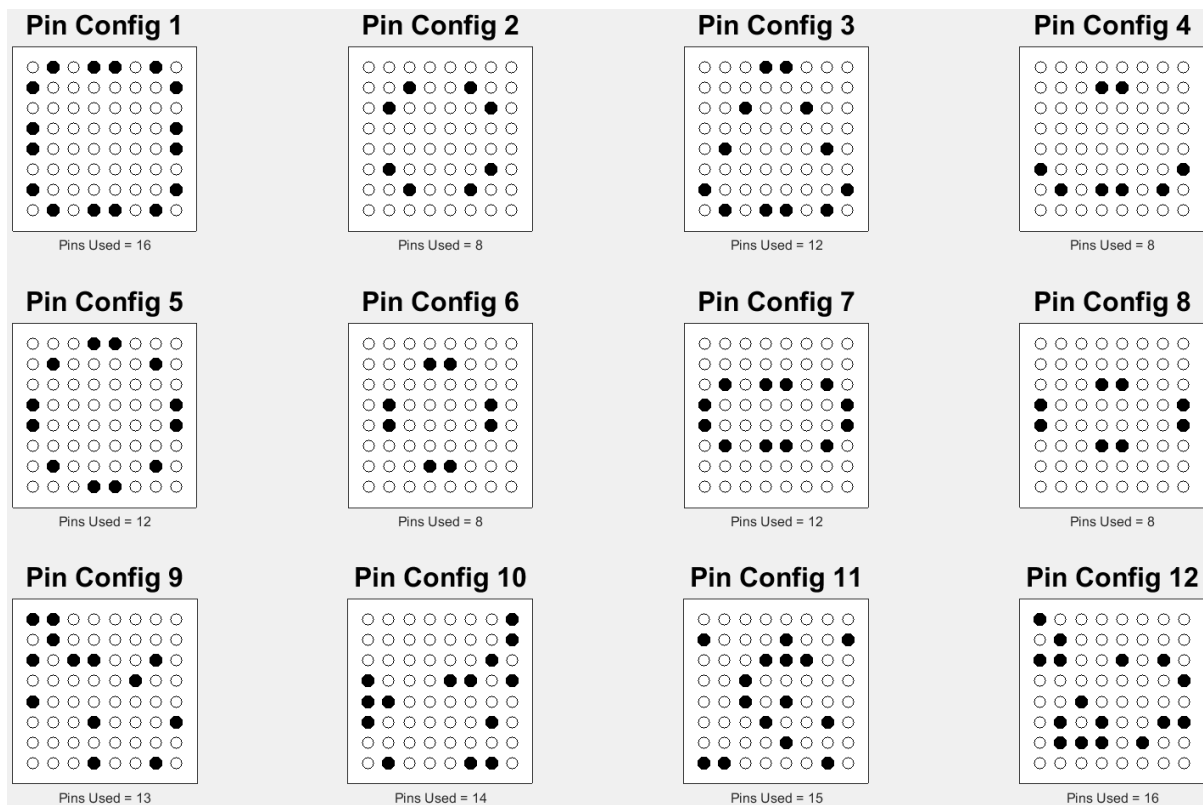


Figure 8.1: Sample set of pin configurations

The first eight pin configurations represent part shapes that correspond to the following shapes:

- 1) Large Square,
- 2) Medium Square,
- 3) Large Triangle,
- 4) Medium Triangle,
- 5) Large Circle,
- 6) Medium Circle,

- 7) Rectangle, and
- 8) Oval.

The last four pin configurations correspond to arbitrarily-shaped parts consisting of 13, 14, 15 and 16 pins, respectively.

The pin configurations lie within the pin range of 8 – 16 pins (as specified in Section 3.4).

The operation times used were randomised within the range of 30 – 90 seconds, based on rough testing in the Lab FxMC. The times used for the sample problem are displayed in Table 8.1. These would be derived from the CAD/CAM software estimation for operation times based on the customer orders (see Section 5.4).

Table 8.1: Operation times

Part number	Fixture reconfiguration time ρ_{ij} (s)	Part processing time τ_{ij} (s)
1	73	83
2	55	49
3	55	42
4	38	55
5	30	31
6	48	70
7	62	89
8	54	78
9	41	38
10	51	42
11	35	64
12	71	72

8.3 Stage I

The pin configurations were compared using Equation (6.3), as described in Figure 6.4. The data was amalgamated into a non-metric distance matrix, as shown in Table 8.2.

Table 8.2: Non-metric distance matrix

Part	1	2	3	4	5	6	7	8	9	10	11	12
1	0	0.883	0.591	0.883	0.591	0.883	0.817	0.760	0.832	0.771	0.859	0.926
2	0.883	0	0.817	0.716	0.817	0.716	0.591	0.716	0.803	0.788	0.844	0.654
3	0.591	0.817	0	0.817	0.716	0.728	0.872	0.817	0.792	0.813	0.832	0.913
4	0.883	0.716	0.817	0	0.728	0.341	0.817	0.716	0.803	0.824	0.812	0.800
5	0.591	0.817	0.716	0.728	0	0.817	0.716	0.591	0.792	0.813	0.904	0.849
6	0.883	0.716	0.728	0.341	0.817	0	0.817	0.716	0.837	0.824	0.812	0.861
7	0.817	0.591	0.872	0.817	0.716	0.817	0	0.184	0.745	0.719	0.796	0.667
8	0.760	0.716	0.817	0.716	0.591	0.716	0.184	0	0.707	0.742	0.773	0.800
9	0.832	0.803	0.792	0.803	0.792	0.837	0.745	0.707	0	0.792	0.846	0.750
10	0.771	0.788	0.813	0.824	0.813	0.824	0.719	0.742	0.792	0	0.830	0.873
11	0.859	0.844	0.832	0.812	0.904	0.812	0.796	0.773	0.846	0.830	0	0.859
12	0.926	0.654	0.913	0.800	0.849	0.861	0.667	0.800	0.750	0.873	0.859	0

The data from Table 8.2 was scaled from twelve-dimensional space to two-dimensional space. Equation (6.4) was used to minimise the stress value to 0.144. Comparing the value obtained against the evaluation table compiled by Sturrock and Rocha [105], the obtained value was within the stated threshold of 0.183 for twelve objects scaled to two dimensions. The resultant 2D map is shown on the left of Figure 8.2.

The k-means++ algorithm [107] was used to initiate the cluster centroids, before the k-means clustering algorithm [106] was used to cluster the data to four groups (for four fixtures). The resultant clusters are shown on the right of Figure 8.2.

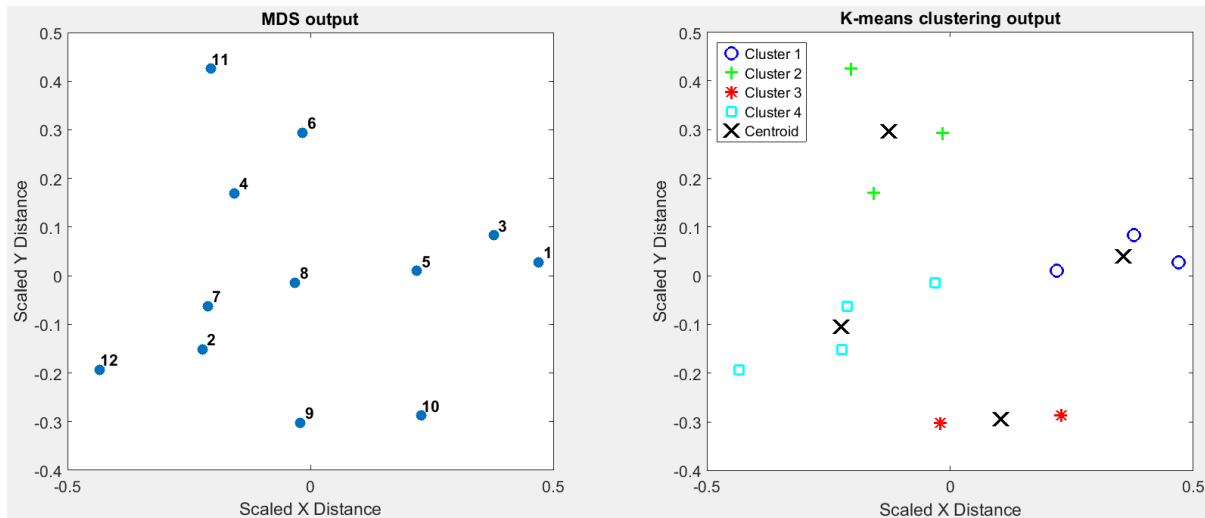


Figure 8.2: MDS plot (left) and k-means output (right)

The goodness of the clusters formed can be evaluated from the silhouette plot [114] shown in Figure 8.3. The silhouette values obtained are mostly above 0.5 and close to 1, with none below zero. This suggests that the clustering procedure successfully grouped similar parts together, such that the reconfiguration effort between members of that group is minimised. The two lower values below 0.5 would necessitate a comparatively greater reconfiguration effort for those parts, due to a lower similarity in relation to their peers. However, the groups formed would have minimised the total dissimilarity between all parts, so that the clusters formed were the best possible. This would increase the likelihood of reconfigurations within those groups requiring minimal time. The sequence in which those reconfiguration efforts can be said to be minimised is only handled in the next stage.

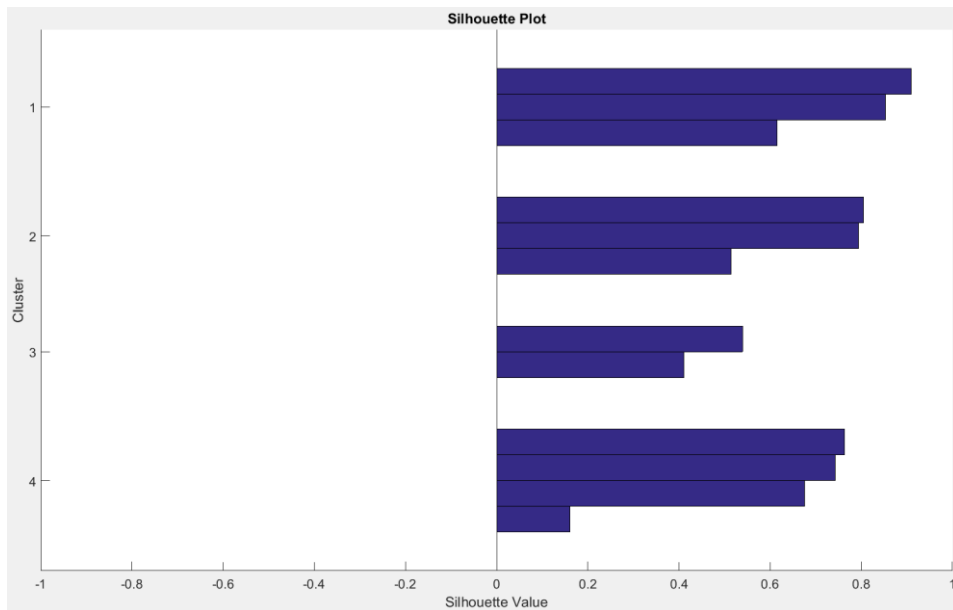


Figure 8.3: Silhouette plot for sample problem

8.4 Stage II

The resultant clusters from Stage I were optimally sequenced in Stage II. Agglomerative hierarchical clustering was used to construct the dendrogram shown at the top of Figure 8.4 to Figure 8.7. Single linkage was used to ensure that the subtrees were formed on a basis of closest pairs. The closeness between objects in this stage refers back to the non-metric distance matrix in Table 8.2 for better accuracy than the MDS map in Figure 8.2. The default dendrogram was then optimally reordered using node-flipping to minimise the total cumulative dissimilarity within each cluster. Figure 8.4, Figure 8.5 and Figure 8.7 demonstrate the efficacy of the optimal reordering, as node-flipping yielded an improved result for all those cases. Node-flipping was not possible in Figure 8.6 due to there being only two objects.

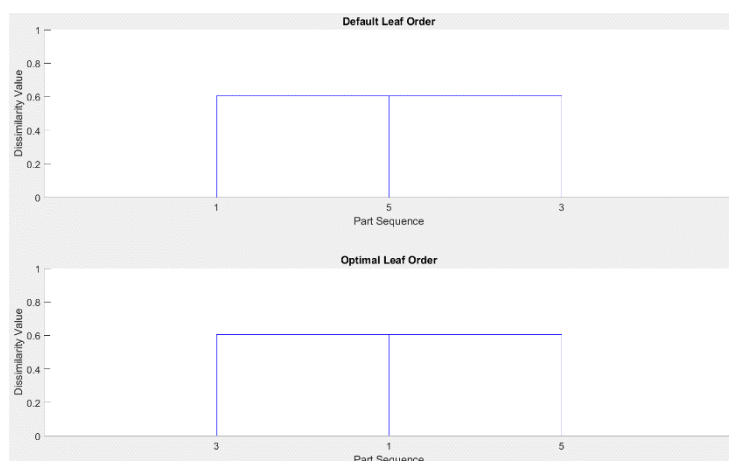


Figure 8.4: Intracluster sequence for Fixture 1

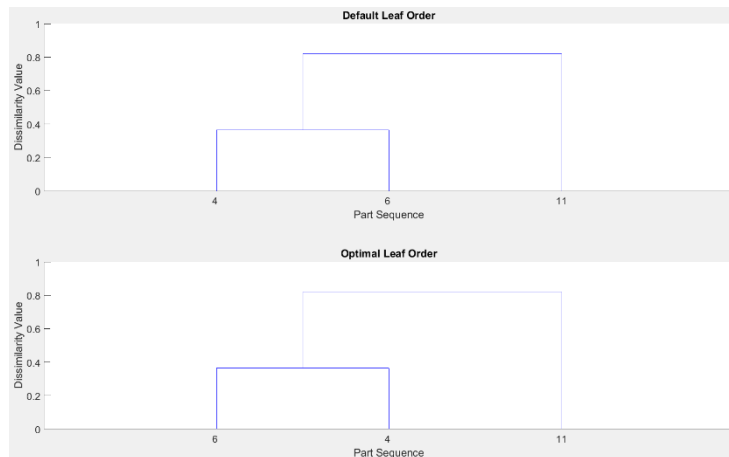


Figure 8.5: Intracluster sequence for Fixture 2

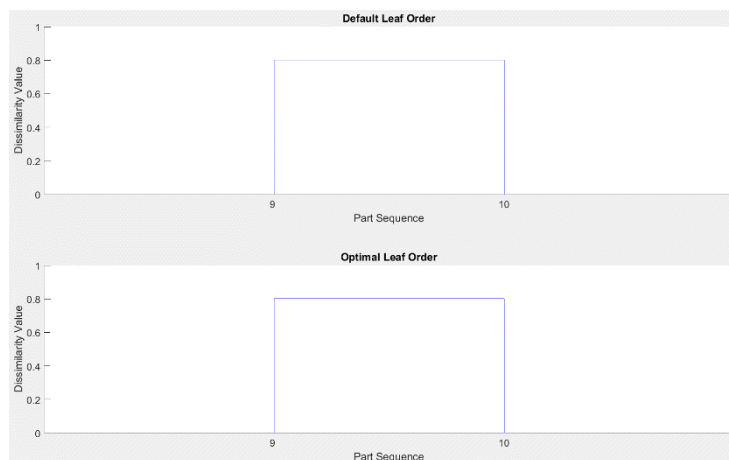


Figure 8.6: Intracluster sequence for Fixture 3

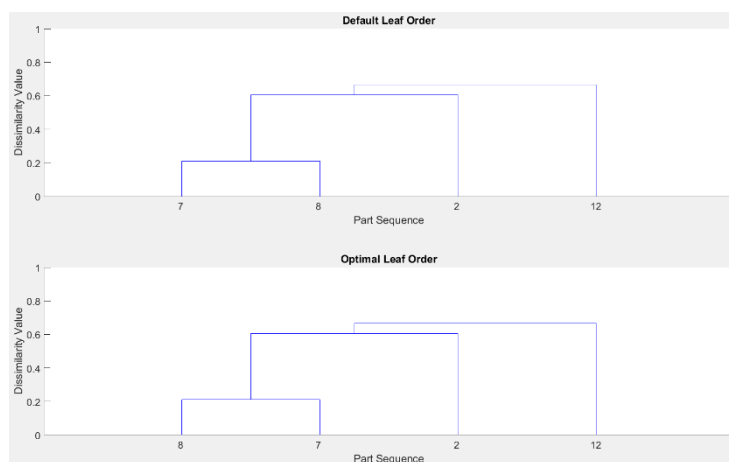


Figure 8.7: Intracluster sequence for Fixture 4

The intracluster sequences generated from this process guaranteed that the dissimilarity between successively reconfigured pin configurations were minimised. This reduced the reconfiguration effort, and thus operation times, for the fixture reconfiguration operations. Minimised fixture reconfiguration times reduce the likelihood of the fixture reconfiguration operations causing a bottleneck in the production system.

8.5 Stage III

The intracluster sequence yielded the matrix used as the input to the MILP model of Stage III. The operation times from Table 8.1 were used in conjunction with this matrix to optimally schedule the fixture reconfiguration and part processing operations, as described by Equations (6.5) to (6.14). The matrix generated is shown in Equation (8.1).

$$\begin{bmatrix} 3 & 1 & 5 & 0 \\ 6 & 4 & 11 & 0 \\ 9 & 10 & 0 & 0 \\ 8 & 7 & 2 & 12 \end{bmatrix} \quad (8.1)$$

The branch and bound algorithm minimised the total idle time generated from the pairs of fixture-part mappings and their operation times. Figure 8.8 shows the graph of convergence for the branch and bound algorithm. The Upper Bound (UB) yielded two feasible solutions, before the growth of the Lower Bound (LB) increased to within range of the UB. The relative gap is shown to be zero, since the algorithm terminated upon establishing that the UB and LB at that node were equivalent, thus warranting the second feasible solution as optimal. This solution was shown to be 116 seconds.

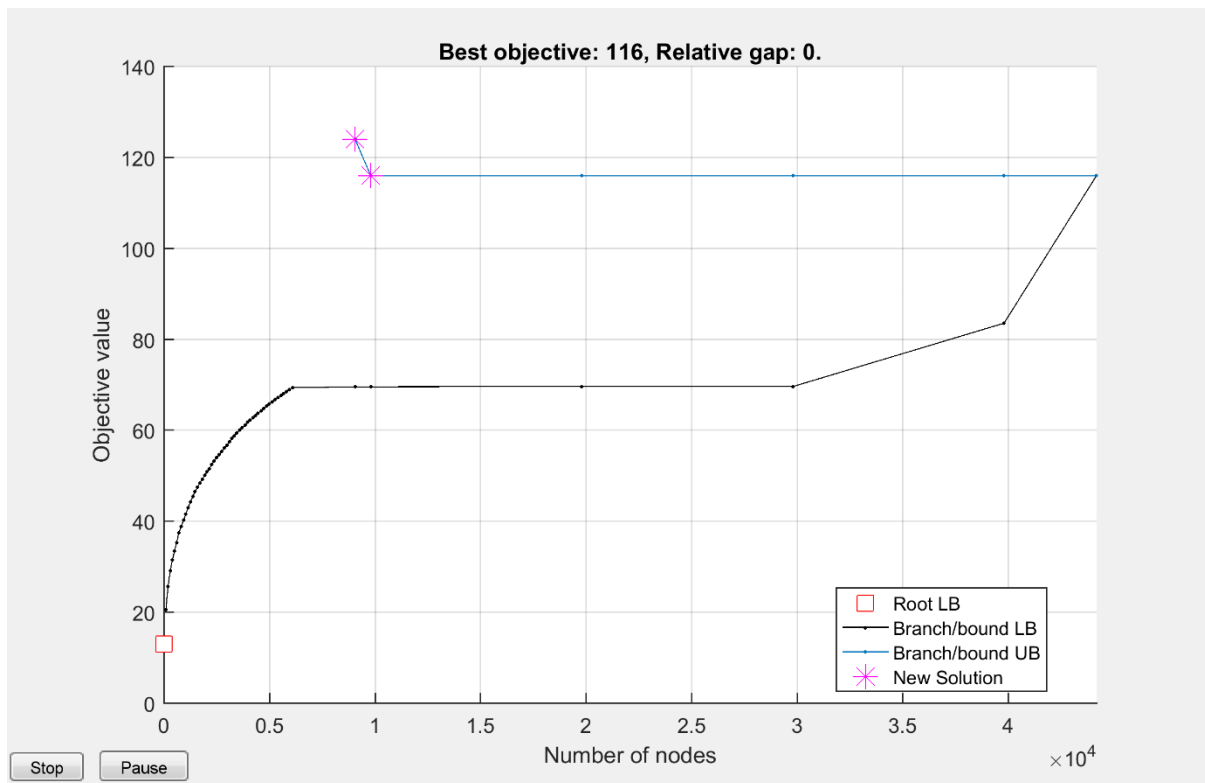


Figure 8.8: Graph of convergence

The resultant schedule, as per the active decision variables Z_{ijklk} , is shown in Table 8.3, together with the synchronous individual idle times.

Table 8.3: Final schedule (optimal)

Fixture i	Intracuster Sequence $j \in I$	Part	Final Sequence k	Idle Time (s)
2	1	6	1	
				16
4	1	8	2	
				23
1	1	3	3	
				1
3	1	9	4	
				0
2	2	4	5	
				7
4	2	7	6	
				16
1	2	1	7	
				28
4	3	2	8	
				2
3	2	10	9	
				12
1	3	5	10	
				4
2	3	11	11	
				7
4	4	12	12	
Total Idle Time (s)				116

8.6 Stage III Heuristic

Alternatively to the MILP model, the Stage III Heuristic could be used to yield the final schedule. The matrix in Equation (8.1) is transposed and operated on as described in Figure 6.12. The resultant schedule is shown in Table 8.4.

Table 8.4: Final schedule (heuristic)

Fixture i	Intracuster Sequence $j \in I$	Part	Final Sequence k	Idle Time (s)
1	1	3	1	
3	1	9	2	1
2	1	6	3	10
1	2	1	4	3
4	1	8	5	29
3	2	10	6	27
2	2	4	7	4
4	2	7	8	7
2	3	11	9	54
4	3	2	10	9
1	3	5	11	19
4	4	12	12	40
Total Idle Time (s)				203

The heuristic generated a scheduled with a total idle time of 203 seconds — 87 seconds greater than the optimal solution.

8.7 Simulation

The schedules derived were simulated with the agent-based simulation. The results are shown in Figure 8.9 and Figure 8.10 for Table 8.3 and Table 8.4, respectively.

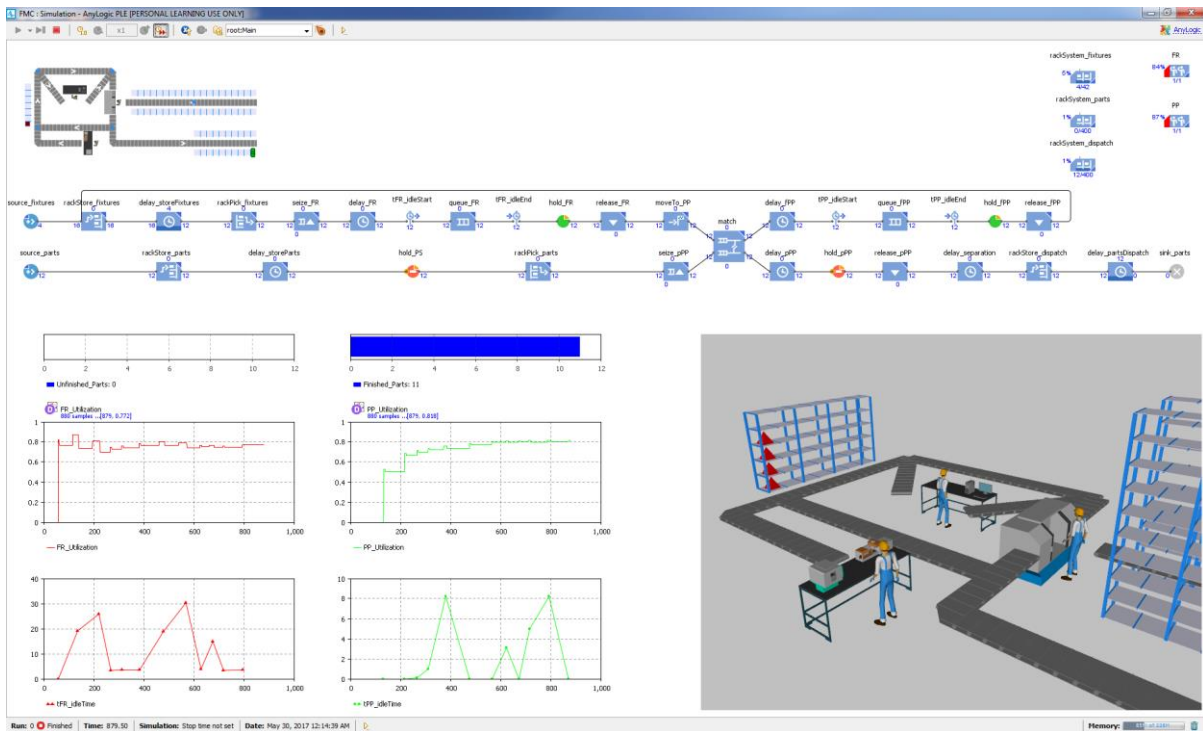


Figure 8.9: Optimal schedule simulation



Figure 8.10: Heuristic schedule simulation

The optimal schedule in Figure 8.9 shows greater utilisation than the schedule derived from the heuristic, as expected due to the minimised total idle time. The greedy approach of the S3H is evident by the lower idle times observed at the start of the simulation in Figure 8.10, which then increase sharply towards the end. The optimal schedule, meanwhile, shows idle time curves with greater variation, which resulted in the total idle time being minimised, instead of the initial idle times only.

8.8 **Summary**

This chapter presented a sample problem, which was solved and presented step-by-step. The results of Stage I and Stage II were shown. The final schedule was then solved with both the MILP model and Stage III Heuristic, with the final result of the MILP model being superior in quality (with respect to the objective function value). Both schedules were then simulated, with observations in utilisation and idle time behaviour displayed and noted. The characteristics observed are further tested, evaluated and elaborated in Chapter 9.

9 Testing and Results

9.1 Introduction

This chapter details the tests conducted on the research outcomes described in Chapter 6 (scheduling method with heuristic) and Chapter 7 (agent-based simulation); as per the sixth and final objective in Section 1.4. Each stage of the scheduling method, the Stage III Heuristic and the agent-based simulation were tested. The findings of the results are reported in the subsequent sections. The tests were conducted on an Intel® Xeon® CPU E3-12710 v3 at 3.50 GHz with 16 GB RAM on a 64-bit operating system. The sections are each decomposed into: aim, methodology, results and conclusion.

9.2 MDS Robustness

9.2.1 Aim

To determine the robustness of the multidimensional scaling algorithm in the application of the research.

9.2.2 Methodology

Kruskal's normalised stress criterion ('stress') was obtained and verified for a variety of test samples. Problem sizes ranging from 10 parts to 100 parts in increments of 10 parts were tested. The parts were based on randomly generated pin configurations. The tests were each repeated ten times using ten different seeds of the MATLAB® random number generator. The Mersenne Twister algorithm [115] was used to generate the random data. The resultant stress value from the MDS procedure was then compared to the evaluation table developed by Sturrock and Rocha [105]. The evaluation table provides a threshold value, which is known as the 1% left-hand-tail cut-off value. If the stress value is above the threshold, there is a greater than 1% probability that the scaled data is without structure and thus random. The evaluation table provides threshold values for scaling from up to 100 objects (100-dimensional data) to the two dimensions required for the k-means clustering algorithm. The MDS procedure was run for 100 iterations for each test, with the lowest stress value generated from that sample set being selected as the final result.

- Test range: 10:10:100 parts
- Repetitions per test: 10
- Criterion: Kruskal's normalised stress

9.2.3 Results

Table 9.1 shows the synthesis of the results compiled in Appendix A.1. The threshold value is the MDS 1% left-hand-tail cut-off value.

Table 9.1: MDS results synthesis

10 parts		20 parts	
Threshold value	0.133	Threshold value	0.279
Average value	0.140	Average value	0.244
Average tolerance %	-5.55	Average tolerance %	12.52
30 parts		40 parts	
Threshold value	0.328	Threshold value	0.352
Average value	0.279	Average value	0.299
Average tolerance %	14.85	Average tolerance %	15.02
50 parts		60 parts	
Threshold value	0.366	Threshold value	0.376
Average value	0.311	Average value	0.319
Average tolerance %	15.14	Average tolerance %	15.20
70 parts		80 parts	
Threshold value	0.384	Threshold value	0.388
Average value	0.327	Average value	0.331
Average tolerance %	14.88	Average tolerance %	14.63
90 parts		100 parts	
Threshold value	0.392	Threshold value	0.396
Average value	0.336	Average value	0.340
Average tolerance %	14.25	Average tolerance %	14.14

9.2.4 Analysis

Scaling from 10 parts (or 10 dimensions) to 2 dimensions produced stress values that fluctuated above and below the threshold stress value of 0.133. The average stress value from the 10 tests was 0.140, which is 5.55 % outside the threshold stress value. The subsequent tests, ranging from 20 parts to 100 parts, resulted in improved robustness of the MDS procedure. The average stress values for these tests were always within their respective threshold stress values. The average tolerance percentages for these tests were mostly around 15 %. The individual stress values for these tests produced very low standard deviations (SD); as such, they were consistently within a very close range of the average value for that dimension.

The results were mostly consistent and within specifications. Scaling from 10 parts was the only test sample that revealed anomalies. The satisfactory results revealed that the two-dimensional maps, which are to be used to cluster parts, adequately represent the original non-metric distance matrices from which they were derived, i.e. there is a less than 1 % probability that the scaled data does not adequately represent the original data. The accuracy of the scaled data is indicated by the stress values themselves. A value greater than zero implies a loss of accuracy in comparison to the original data. The stress values were expected to be above zero, which was due to: the data being scaled from very high dimensions to only two dimensions, and; the original data being non-metric, which means that there can be no true representation of the data in real space. The stress values were consistently above 0.1, which affected the accuracy of the results. However, as shown by the comparison to the evaluation table, most of the results were satisfactory.

Scaling from lower dimensions (below 20 parts) would hamper the results more so than higher dimensions. The results were, at most, 24.29 % above the threshold value. It should be noted that the threshold value represents a probability distribution, which means that a stress value above the threshold does not necessarily represent a dataset without structure; it only means that there is a greater probability that it does. The standard deviation for the 10-parts test sample was also much higher than that of any

other test sample, due to the fluctuation of the results above and below the threshold value; this means that scaling from lower dimensions does not guarantee a stress value above the threshold.

9.2.5 Conclusion

The robustness of the MDS procedure is high for scaling from 20 dimensions (or 20 parts) and above, with consistent results that were within specifications. Scaling from lower dimensions was not as satisfactory; as such, monitoring of the results would be required to ensure that the dataset for a given test is not above the threshold value. It should also be noted that the application of the research is for mass customisation production systems; therefore, the performance of the procedure for higher part numbers is of greater significance. The accuracy of the data is still inherently compromised by the scaling procedure itself, as made apparent by the individual stress values being consistently above zero.

9.3 Silhouette Values (Constant Parts, Variable Fixtures)

9.3.1 Aim

To investigate the effect of increasing fixture quantity for a constant number of parts on the goodness of clusters formed by the k-means clustering algorithm in the application of the research.

9.3.2 Methodology

Silhouette values were obtained from the clustering results, as described by Rousseeuw [114]. The average silhouette value was determined for each dataset. A part quantity of 100 parts was used for the tests. The fixture quantity was increased from 2 fixtures to 10 fixtures in increments of 2 fixtures. The Mersenne Twister algorithm was used to randomise pin configurations for the 100 parts. The test was repeated three times using three different seeds of the MATLAB® random number generator. The k-means algorithm is an iterative algorithm; therefore, the algorithm was run for *parts*fixtures* iterations, which grew in relation to the problem size.

- Test range: 2:2:20 fixtures; 100 parts
- Repetitions per test: 3
- Criterion: Silhouette values

9.3.3 Results

Table 9.2 shows the synthesis of the results compiled in Appendix A.2 (where ‘f’ represents fixtures).

Table 9.2: Silhouette values results synthesis

Sample 1										
	2 f	4 f	6 f	8 f	10 f	12 f	14 f	16 f	18 f	20 f
Average	0.469	0.516	0.5205	0.5036	0.4955	0.4958	0.5183	0.5146	0.5055	0.5201
Median	0.5313	0.5493	0.532	0.5021	0.5056	0.53	0.5438	0.5505	0.5479	0.5498
SD	0.2286	0.2321	0.2399	0.2253	0.233	0.2477	0.217	0.219	0.2518	0.2281
Minimum	-0.1101	0.0257	-0.1302	-0.0464	-0.0965	-0.06	0.054	-0.0846	-0.2839	-0.2686
Maximum	0.7466	0.8368	0.8502	0.8383	0.8489	0.8458	0.8579	0.8507	0.8694	0.8709

Table 9.2: Silhouette values results synthesis (continued...)

Sample 2										
	2 f	4 f	6 f	8 f	10 f	12 f	14 f	16 f	18 f	20 f
Average	0.4703	0.5148	0.512	0.4977	0.4978	0.5072	0.5314	0.533	0.534	0.5438
Median	0.5238	0.5427	0.5718	0.5153	0.5543	0.5489	0.5553	0.5545	0.5883	0.575
SD	0.2247	0.2392	0.2448	0.2258	0.2287	0.2298	0.2309	0.2189	0.2347	0.2393
Minimum	-0.1683	-0.0279	-0.0487	0.0075	-0.0899	0.0637	-0.0745	-0.0813	-0.1019	-0.1923
Maximum	0.7549	0.8241	0.8326	0.8662	0.8553	0.8416	0.8551	1	1	1
Sample 3										
	2 f	4 f	6 f	8 f	10 f	12 f	14 f	16 f	18 f	20 f
Average	0.4923	0.5501	0.4885	0.5159	0.5019	0.4999	0.4905	0.502	0.513	0.5024
Median	0.5464	0.623	0.5386	0.5086	0.5322	0.541	0.5563	0.5202	0.5315	0.5044
SD	0.2189	0.2298	0.2349	0.2225	0.2291	0.2228	0.2489	0.2165	0.1946	0.2076
Minimum	-0.212	0.0126	0.0133	-0.0318	-0.0759	-0.1065	-0.0986	0.0386	0.0739	-0.0564
Maximum	0.7566	0.8689	0.8225	0.8639	0.8473	0.8451	0.8521	0.834	0.8911	0.8887

Figure 9.1 to Figure 9.3 display the average silhouette values obtained by increasing the fixture quantity from 2 to 20 fixtures for 100 parts (displaying the trend observed for each of the samples tested).

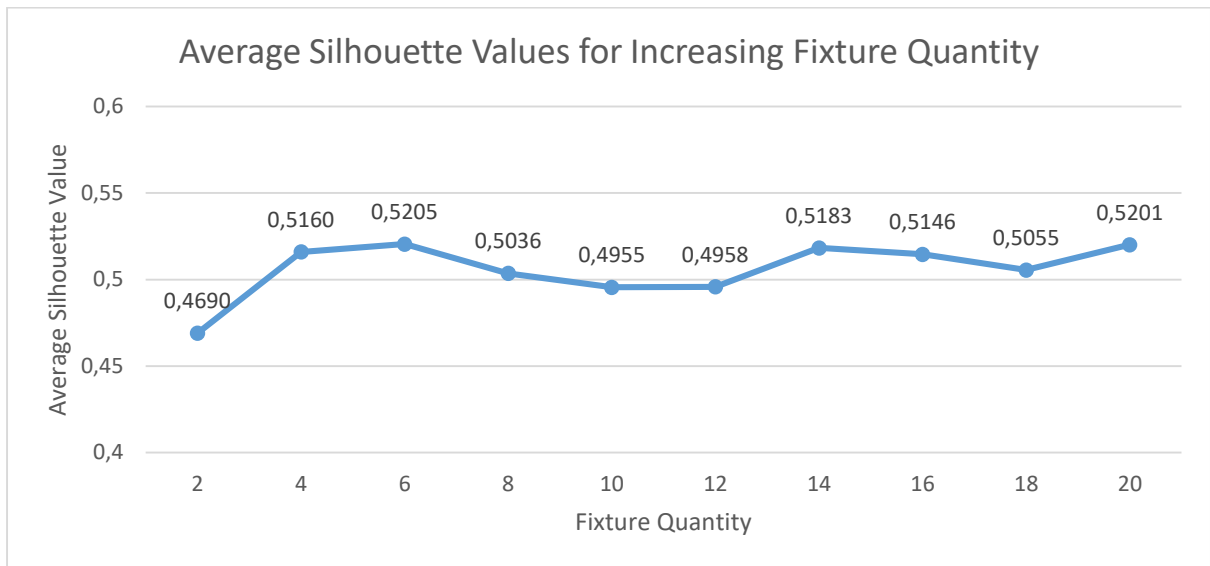


Figure 9.1: Average Silhouette Values for Increasing Fixture Quantity (Sample 1)

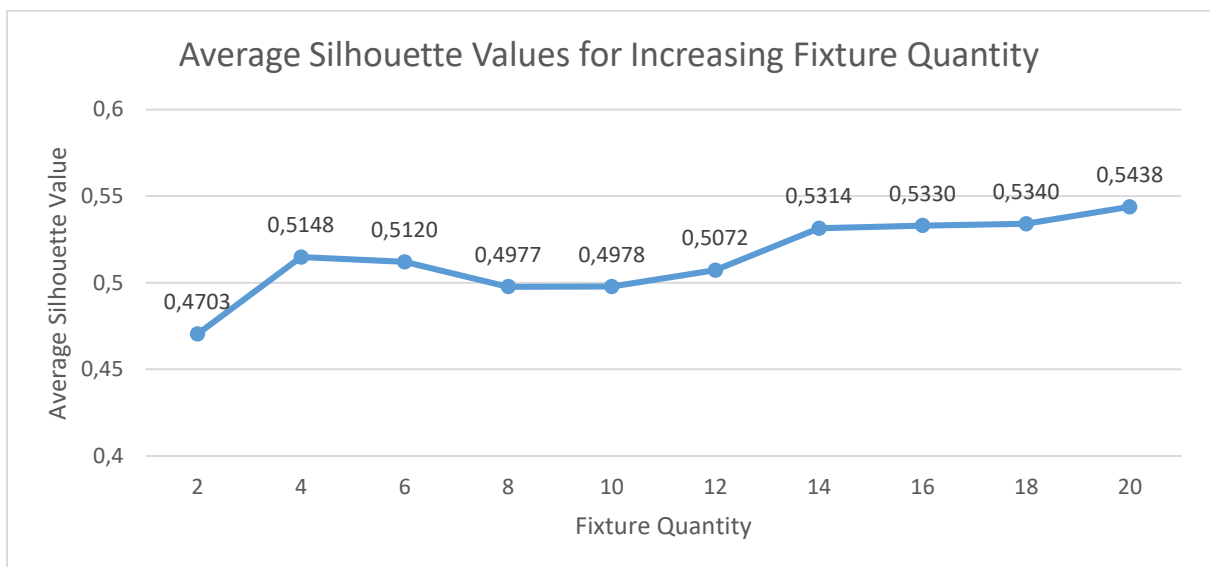


Figure 9.2: Average Silhouette Values for Increasing Fixture Quantity (Sample 2)

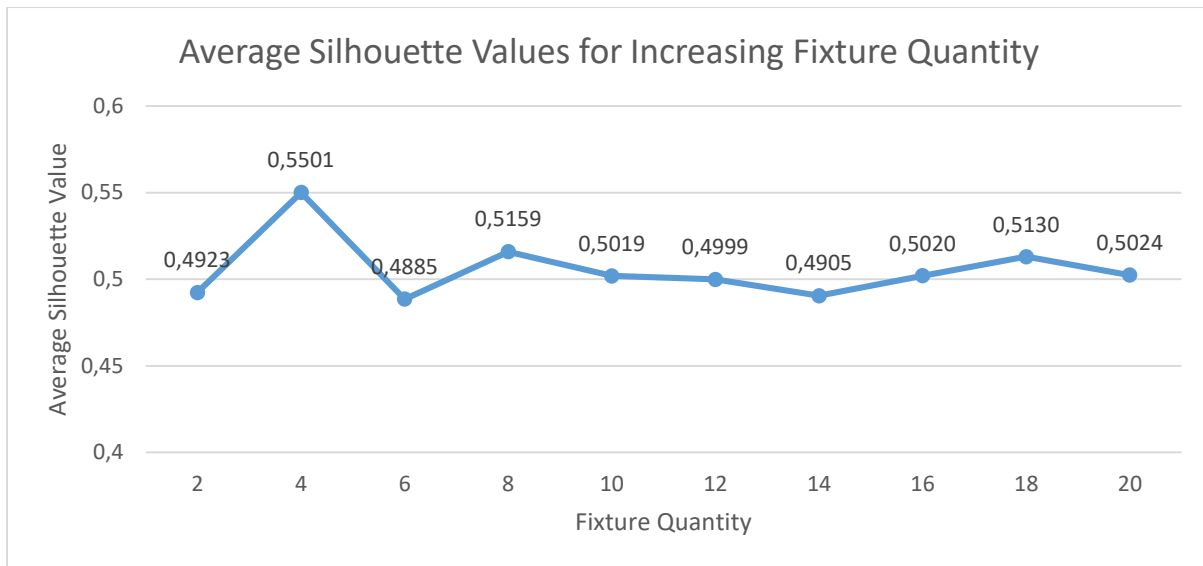


Figure 9.3: Average Silhouette Values for Increasing Fixture Quantity (Sample 3)

9.3.4 Analysis

Figure 9.1 to Figure 9.3 did not display any discernible trend, apart from a consistent increase in average silhouette values for 4 fixtures in comparison to 2 fixtures. It was noted that the cluster size fail-safe heuristic (see Section 6.7), which prevents cluster size distribution from yielding infeasible solutions in Stage III, was utilised for the 2-fixture scenarios in every test sample; this would result in a decrease of the silhouette values produced, which clarifies why the data points for these circumstances were consistently the lowest. The cluster size fail-safe heuristic was not required for any of the other fixture quantities. The trend observed after 2 fixtures varies in a random fashion, with no clear indication of what fixture quantity would be optimal with regards to the goodness of clusters generated.

The test also provided information on the goodness of the clusters generated. The average silhouette values were reasonably high (around 0.5), with the medians predominantly higher than the averages, thus indicating that the majority of individual silhouette values were in the higher segment of the silhouette value range of $[-1, 1]$. Negative silhouette values were rarely observed, with only 50 such instances appearing in the silhouette plots (see Appendix A.2.1); these instances constituted 1.67 % of the total silhouette values obtained. The lowest of the negative silhouette values was -0.2839, which is favourably closer to 0 than -1.

9.3.5 Conclusion

No observable trend was observed by increasing the fixture quantity for a constant number of parts; however, two fixtures for such large-sized problems (as the 100 part problem used for this test) should be avoided to either prevent infeasible solutions in Stage III or maximise the goodness of the clusters formed. The fail-safe heuristic was only required for the 2-fixture problem, thus validating that evenly-sized clusters would not be required to ensure feasibility in Stage III. The silhouette values generated were deemed sufficiently high, and negative values were sufficiently scarce; therefore it can be concluded that the clusters generated were sufficiently good.

The goodness of the clusters formed validate the dissimilarity measure used (Equation (6.3)), as the clusters were clear and unambiguous. The high silhouette values obtained imply that the pin configurations sharing the same fixture base plate are highly similar to each other. As described in Section 6.7, the dissimilarity measure favours minimisation of pin manipulations; therefore, it can be rationalised that the clusters generated promote quick fixture reconfigurations, thus minimising individual operation times and total makespan.

9.4 Cluster Reordering Effectiveness

9.4.1 Aim

To investigate the degree of improvement in total cumulative pairwise dissimilarity obtained by reordering the default leaf order for the intracluster sequence.

9.4.2 Methodology

The clusters formed from the k-means clustering algorithm are placed into groups in an initial (arbitrary) order. The agglomerative hierarchical clustering algorithm with single linkage was used to produce the default order. The optimal leaf order algorithm was used to produce the optimal order from the default order. The total cumulative pairwise dissimilarity was recorded by calculating the summation of pairwise dissimilarities from the non-metric distance matrix, as per the order evaluated (initial, default or optimal). The improvement obtained was recorded as a percentage. The test was conducted using a constant 100 parts. The datasets were evaluated for 4, 10 and 16 fixtures each. The Mersenne Twister algorithm was used to randomise pin configurations for the 100 parts. The test was repeated ten times using ten different seeds of the MATLAB® random number generator.

- Test range: 4, 10, 16 fixtures; 100 parts
- Repetitions per test: 10
- Criterion: Total cumulative pairwise dissimilarity improvement percentage

9.4.3 Results

Table 9.3 shows the synthesis of the results compiled in Appendix A.2.1 for the comparison of the optimal (final) order to the initial order only.

Table 9.3: Cluster reordering results synthesis

Sample 1			
	4 fixtures	10 fixtures	16 fixtures
Average (%)	11.47	8.90	7.75
SD (%)	1.99	3.37	4.45
Minimum (%)	9.08	3.49	0
Maximum (%)	13.84	15.06	15.66
Sample 2			
	4 fixtures	10 fixtures	16 fixtures
Average (%)	10.06	8.53	6.34
SD (%)	0.86	3.71	4.04
Minimum (%)	9.02	3.61	1.31
Maximum (%)	11.09	15.68	17.99

Table 9.3: Cluster reordering results synthesis (continued)

Sample 3			
	4 fixtures	10 fixtures	16 fixtures
Average (%)	11.04	8.83	7.03
Standard deviation (%)	1.97	4.40	4.21
Minimum (%)	8.19	3.57	0
Maximum (%)	12.73	17.65	16.24
Sample 4			
	4 fixtures	10 fixtures	16 fixtures
Average (%)	12.65	11.28	8.56
SD (%)	1.18	2.30	3.96
Minimum (%)	10.98	6.94	0
Maximum (%)	13.76	14.38	16.53
Sample 5			
	4 fixtures	10 fixtures	16 fixtures
Average (%)	11.69	9.05	5.85
SD (%)	3.69	2.76	2.92
Minimum (%)	7.65	5.54	1.32
Maximum (%)	16.32	14.32	12.15
Sample 6			
	4 fixtures	10 fixtures	16 fixtures
Average (%)	12.03	10.02	7.39
SD (%)	1.58	4.53	3.85
Minimum (%)	9.67	5.22	0
Maximum (%)	12.92	18.34	14.26
Sample 7			
	4 fixtures	10 fixtures	16 fixtures
Average (%)	14.26	9.84	8.96
SD (%)	1.02	2.28	3.78
Minimum (%)	12.79	5.60	3.01
Maximum (%)	15.13	13.77	19.79
Sample 8			
	4 fixtures	10 fixtures	16 fixtures
Average (%)	12.14	8.79	6.00
SD (%)	3.36	3.07	5.38
Minimum (%)	10.01	3.69	0
Maximum (%)	17.14	14.79	21.11
Sample 9			
	4 fixtures	10 fixtures	16 fixtures
Average (%)	14.34	9.54	7.79
SD	2.26	2.56	4.47
Minimum (%)	12.31	5.35	0
Maximum (%)	17.01	13.98	16.98
Sample 10			
	4 fixtures	10 fixtures	16 fixtures
Average (%)	12.25	9.26	9.03
SD (%)	1.49	3.03	4.84
Minimum (%)	11.15	5.42	1.53
Maximum (%)	14.39	15.57	20.09

Figure 9.4 displays the trends observed for the average improvement percentage for increasing fixture quantity for each of the test samples.

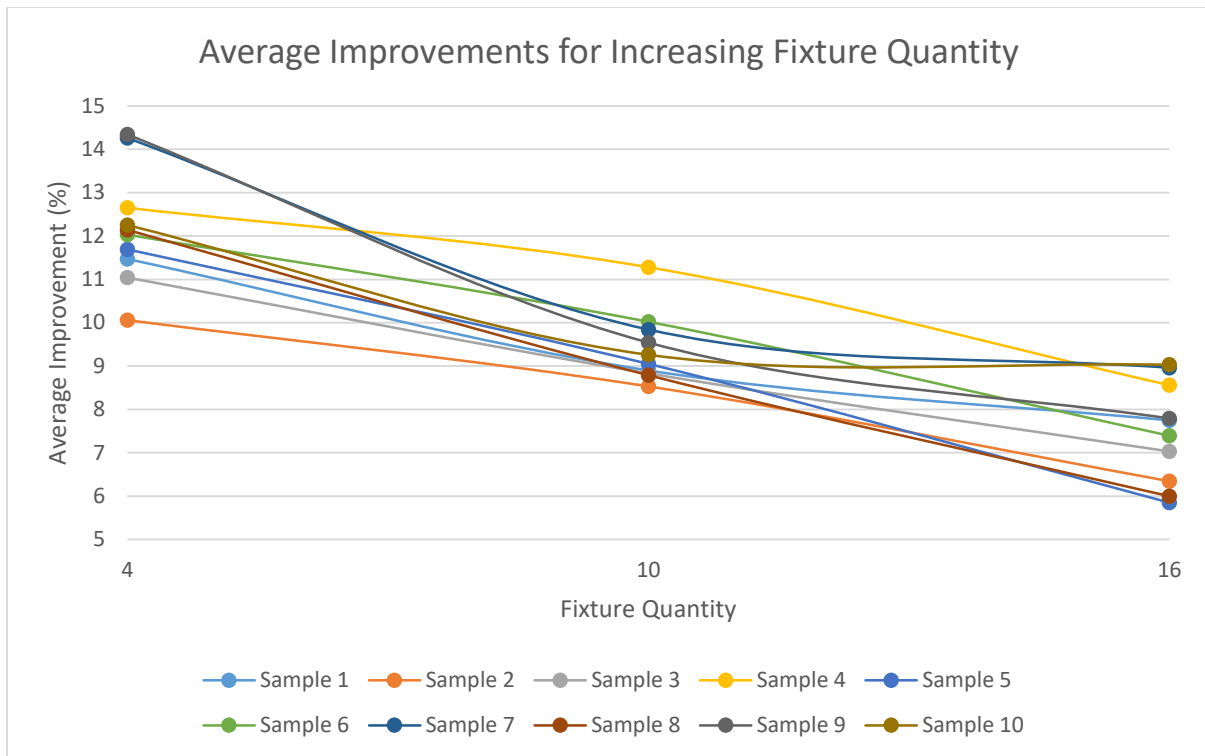


Figure 9.4: Average Improvement for Increasing Fixture Quantity

9.4.4 Analysis

Table 9.3 shows a consistent improvement in the total cumulative pairwise dissimilarity, apart from the rare instances where the initial cluster order was incidentally the same as the optimal order. The improvements observed ranged from 1.32 % to 21.11 % (apart from the 0 % improvement observations). The results prove the effectiveness of agglomerative hierarchical clustering with single linkage and the optimal leaf order algorithm in the application of the research, by consistently improving the intracluster sequence when it was possible.

Figure 9.4 shows the trends observed when comparing the average improvement percentages for increasing fixture quantity. The ten samples exhibited a consistent decrease in average improvement percentage for increasing fixture quantity. The behaviour of the decrease varied for the samples; cases of logarithmic, exponential and linear decreases were observed. The cause of the trend is most likely due to the reduced cluster sizes that originate from an increase in clusters; this would result in fewer objects from which the summation of pairwise dissimilarities is calculated, and fewer node flipping options available to the algorithm. Despite the decreasing average improvement percentage for the larger samples, some of the highest maximum improvement for individual clusters were observed in some of these cases.

9.4.5 Conclusion

Agglomerative hierarchical clustering with single linkage and the optimal leaf order algorithm successfully reduced the total cumulative pairwise dissimilarities for the initial clusters when it was possible, i.e. when the initial order was not already in the optimal order. Improvements of over 20 % were noted for some clusters. The trends for increasing fixture quantity were observed for the ten

samples. It can be concluded that increasing fixture quantity decreases the average improvement percentage per cluster, the trend of which is variable.

The results mean that the intracluster order generated from Stage II is an improvement on the initial order generated in Stage I. The improvement results in a reconfiguration order on each fixture that minimises the pin manipulations required, thus decreasing the fixture reconfiguration time for operations on that fixture. The reduction in fixture reconfiguration time implicitly decreases the total makespan.

9.5 MILP Model Behaviour

9.5.1 Aim

To examine the behaviour and performance of the mixed integer linear programming model for a variety of fixture-part combinations.

9.5.2 Methodology

Rough testing revealed that the MILP model solver could not reliably handle the computational expense of problem sizes that consisted of more than 12 parts. As such, various fixture-part combinations were created within the available range, ensuring that the quantity of fixtures were, at most, half that of the parts. The fixture reconfiguration times and part processing times were each randomised within a range of 30 – 90 seconds using the Mersenne Twister algorithm. The operation time range was determined from rough testing and estimation from the Lab FxMC (Chapter 4). The fixture-part combinations were each tested three times using three different seeds of the MATLAB® random number generator. The tests used a different seed of the random number generator for every run, so that the behaviour of the MILP model could be comprehensively investigated. The MATLAB® MILP solver (*intlinprog*) was used to solve the problems by means of the branch and bound algorithm. Termination of the solver was achieved when the lower bound and upper bound of the B&B solver converged, thus signifying the solver's arrival at the optimal solution.

- Test range: 4:1:12 parts; $2 \leq \text{fixtures} \leq 0.5 * \text{parts}$
- Repetitions per test: 3
- Criteria: Convergence, and solver characteristics & performance

9.5.3 Results

The results presented are based on Appendix A.3.1. See Figure A.61 to Figure A.135 for all graphs of convergence.

Figure 9.5 shows the increase in variables for the MILP model when the fixture quantity was increased for a constant number of parts. The fixture-part combinations displayed were extracted from the 75 test runs detailed in Table A.24.

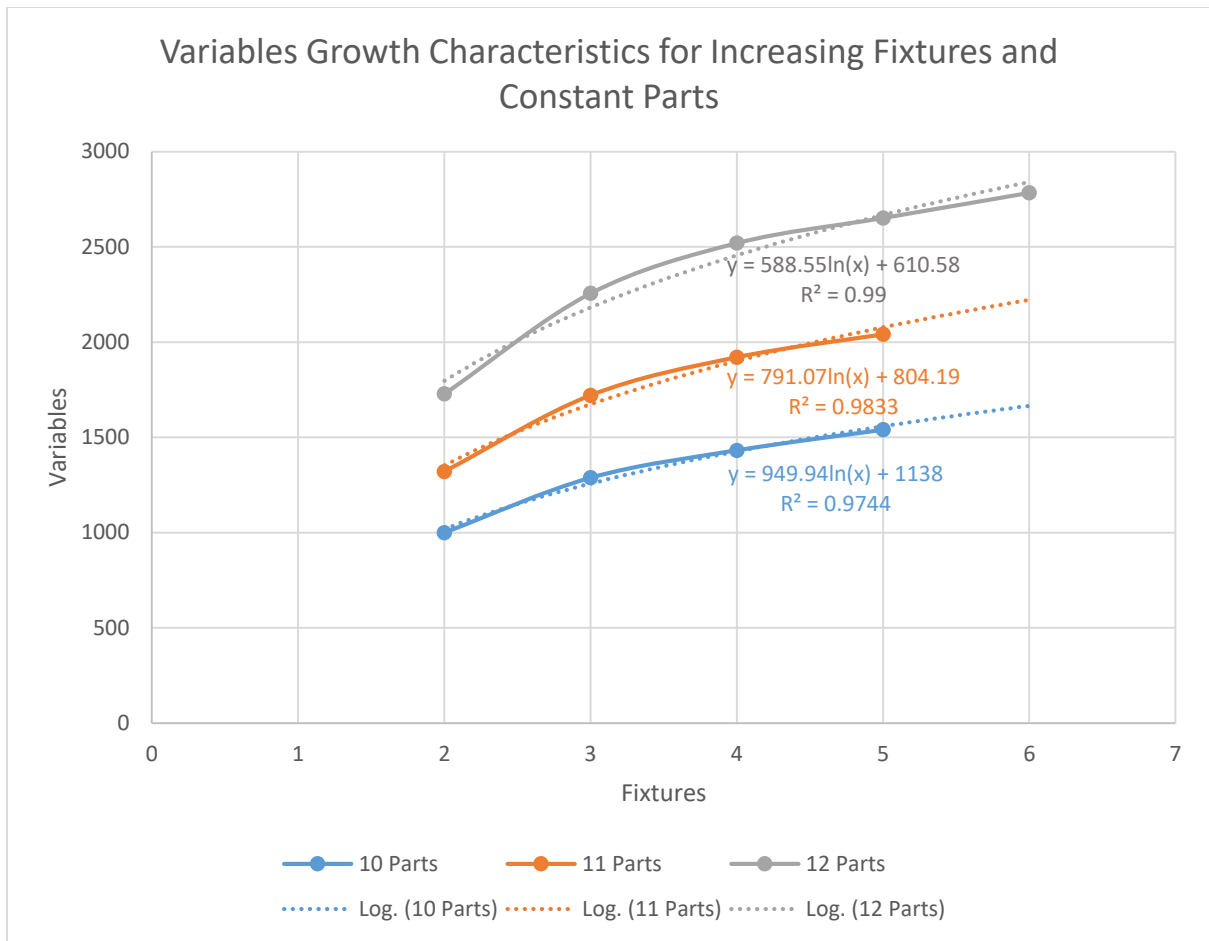


Figure 9.5: Variables Growth Characteristics for Increasing Fixtures and Constant Parts

Figure 9.6 shows the increase in variables for the MILP model when the part quantity was increased for a constant number of fixtures. The fixture-part combinations displayed were also extracted from the 75 test runs detailed in Table A.24.

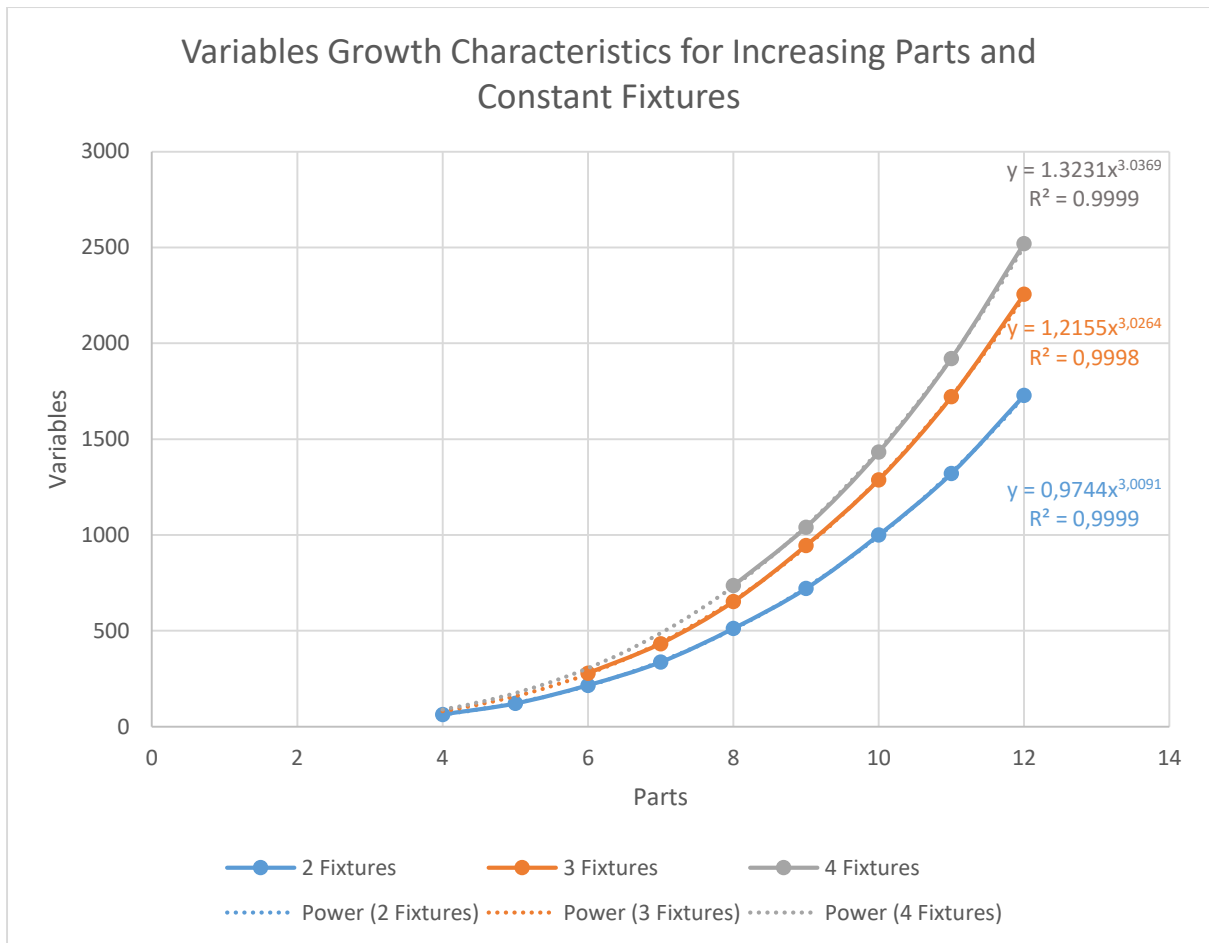


Figure 9.6: Variables Growth Characteristics for Increasing Parts and Constant Fixtures

Figure 9.7 shows the number of nodes explored to solve each problem, in comparison to the number of variables that those problems consisted of.

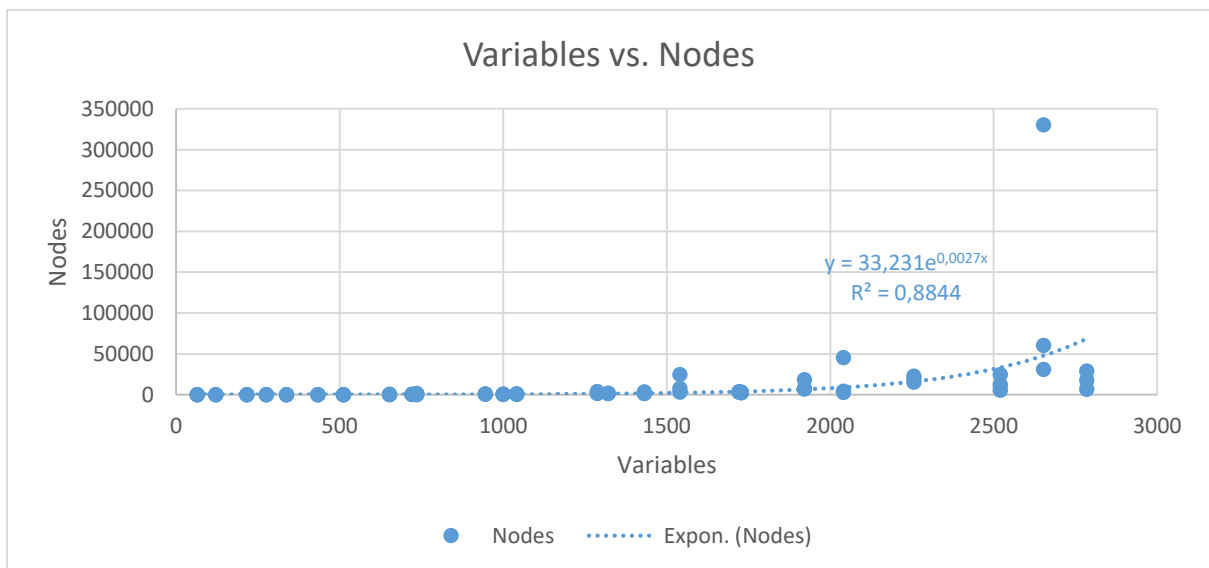


Figure 9.7: Nodes Explored in relation to Variables

Figure 9.8 shows the solution time required to solve each problem, in comparison to the number of variables that those problems consisted of.

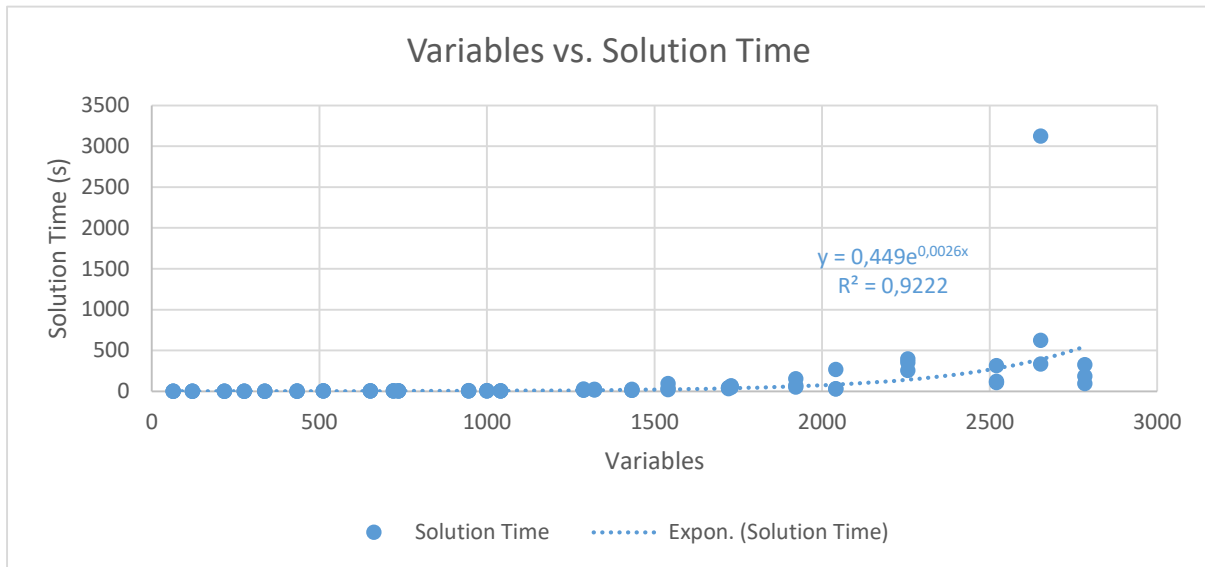


Figure 9.8: Solution Time required in relation to Variables

Figure 9.9 shows the solution time required to solve each problem, in comparison to the number of nodes explored to solve each of those problems.

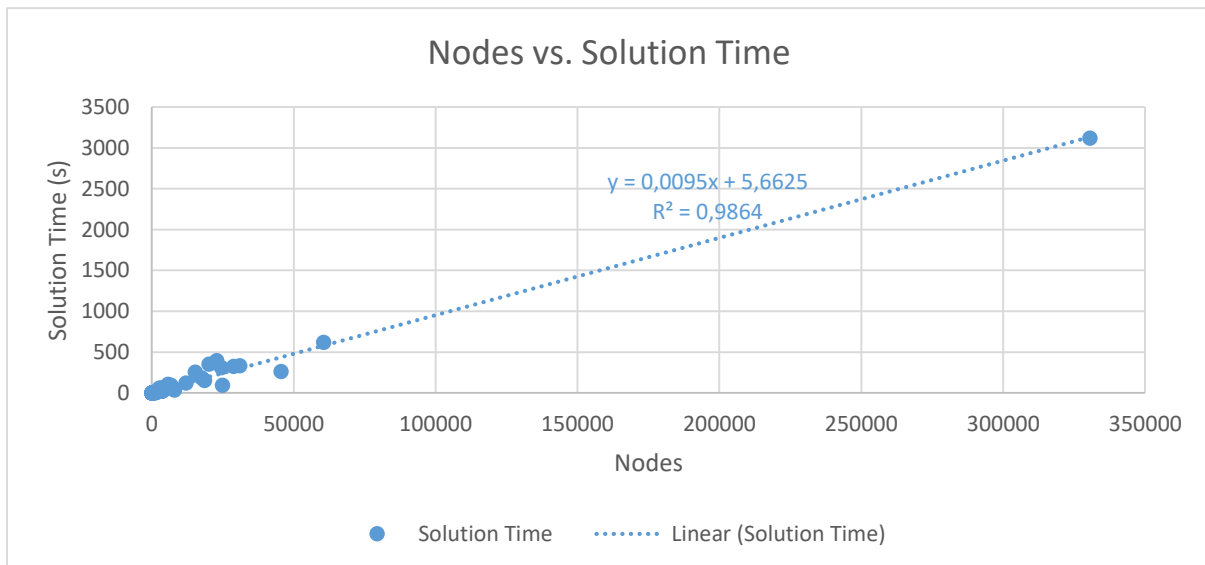


Figure 9.9: Solution Time in comparison to Nodes Explored

9.5.4 Analysis

The results presented in Appendix A.3.1 reveal that the MILP solver converged to a solution for every case that was tested, i.e. a 100 % success rate for the problem range was achieved. Relative gaps of < 0.1 , as per the assigned solver settings, were observed for 27 cases (36 % of the test samples). The relative gaps resulted in absolute gaps that did not exceed 0.2 for most cases, apart from a value of 0.5 for one of the 12-part/2-fixture problems. The gaps, which were ≤ 0.5 , were deemed acceptable due to the integer values that were used for the operation times, meaning that the final result would be the

same as that achieved if the relative gap was zero. The relaxed settings allowed for reduced solution times to be achieved when possible, due to the reduction in nodes explored.

An average of 1.867 feasible solutions were found for the problems tested; no more than 5 feasible solutions were found for any problem, while 1 feasible solution was the most frequent outcome. Rough testing revealed that using the ‘most fractional’ branch rule produced the optimal solution by exploring fewer nodes than the ‘max fun’ option (as mentioned in Section 6.9.3); this would explain why minimal feasible solutions were found, with the benefit of finding the optimal solution in reduced time.

Constraint violations were observed in three cases; the order of magnitude of these violations were no greater than 10^{-15} , which is negligible in comparison to the parameters of the problem (order of magnitude 10^0). Therefore, these results were deemed satisfactory and the solutions regarded as optimal.

Figure 9.5 shows the trend lines for each case (number of parts) that was investigated. The coefficients of determination (R^2) confirmed that the growth characteristic of variables when fixtures were increased was logarithmic. The logarithmic growth is shown to increase for increasing part quantities; the sharpest increase in the number of variables is observed for 12 parts, whereas the 10-part plot is shown to be almost linear. The results reveal that increasing the fixture quantity for a constant number of parts causes the variable size to grow, albeit very slowly.

Figure 9.6 shows the trend lines for each case (number of fixtures) that was investigated. The coefficients of determination (R^2) confirmed that the growth characteristic of variables when parts were increased was polynomial, roughly to the power 3. The rate of polynomial growth is shown to increase slightly for increasing fixture quantities. The equations generated from the graphs confirm the observation of sharper increases for greater fixture quantities via the slightly increasing values of the exponents (3.0091, 3.0264 and then 3.0369). The results reveal that increasing the part quantity for a constant number of fixtures causes the variable size to grow very quickly. It is suggested that it is this characteristic that prevented the MATLAB® MILP solver from solving problems consisting of more than 12 parts.

Figure 9.7 shows a large distribution of the data points. An outlier is observed for one of the 12-part/5-fixture problems; the solution to this problem required the exploration of a far greater number of nodes than its counterparts. The general distribution of the data was notably varied, even when disregarding the outlier. The trend line was constructed with a coefficient of determination (R^2) of 0.8844, which is not ideal, but deemed sufficiently high to recognise the general trend of the observation. The trend observed was that of exponential growth, whereby the nodes explored to find the solution for larger problems (in term of number of variables) grew at a sharply increasing rate; this is only a general trend, as the variation along the trend line is significant. Therefore, there is no definite correlation between the number of variables of a problem and the nodes required to be explored by the branch and bound algorithm to solve it; however, a broad trend of exponential increase is observed, such that solving larger-sized problem would be expected to take longer to solve (with an extent of uncertainty).

Figure 9.8 displays similar characteristics to Figure 9.7. An exponential trend was observed, with a higher coefficient of determination than that of Figure 9.7. The outlier from Figure 9.7 is also detected

in Figure 9.8, with the other data points being similarly scattered as in that previous figure. It can be concluded that the general trend of solution time increase in comparison to problem size (in terms of number of variables) is an exponential increase; however, the degree of uncertainty detected in Figure 9.8 also applies here.

The relationship between nodes explored and solution time was investigated, due to the similarities observed between nodes explored and solution time required when compared to the number variables. Figure 9.9 shows that a linear relationship exists between the solution time and nodes explored. The trend line is shown to agree with the previously observed outlier as well. It can be concluded that the solution time relies on the number of nodes explored to find the solution. The number of nodes required varies greatly with each problem, which means that accurately predicting the solution time for a problem is impracticable. However, the trends observed in Figure 9.7 and Figure 9.8 do suggest an exponential increase in nodes explored, and thus solution time; this agrees with the expected behaviour of the branch and bound algorithm solution time, which is said to increase exponentially with the size of a problem [113].

9.5.5 Conclusion

The optimal solution was found for the problem range tested, verified by the convergence of the lower and upper bounds of the branch and bound solver (see Appendix A.4.1). Non-idealities in some of the results were discussed and it was deduced that the solutions were optimal nonetheless, given the parameters used.

The number of variables was found to increase logarithmically for increasing fixtures. The number of variables was found to increase polynomially for increasing parts, roughly to the power 3. The increase in both nodes and solution time in comparison to number of variables was found to be exponential. The relationship between nodes and solution time was verified to be directly proportional.

The sharp increase in variables for increasing fixtures, together with the exponential increase in nodes and solution time for increasing variables, insinuates why the solver was unable to solve problems with part quantity greater than 12. The inability was likely due to the high computational expense required to solve such problems to optimality.

9.6 Heuristic Solution Quality

9.6.1 Aim

To compare the goodness of solutions generated by the S3H with the equivalent (optimal) MILP solutions, and their solution times thereof.

9.6.2 Methodology

A selection of the problems from Section 9.5 were solved using the Stage III Heuristic. The problems selected were those that produced evenly-sized clusters, simply because it was much quicker to duplicate the parameters required for implementing these problems for the S3H. The problem dataset yielded twelve fixture-part combinations that satisfied the criterion, each of which were repeated three

times as per the seeds used for the MILP problems. The resultant total idle times were compared to inspect the degree of sub-optimality of the S3H in comparison to the optimal solutions. The solution times to obtain the solutions were also compared thereafter.

- Test range: Selected problems from Section 9.5
- Repetitions per test: 3
- Criterion: Total idle time, Solution time

9.6.3 Results

Figure 9.10 displays the discrepancies that existed between the total idle times generated by the Stage III Heuristic in comparison to the optimal solutions generated by the MILP model for the same problems. The specific total idle times are shown in the table incorporated below the graph. The results were generated from Appendix A.5.

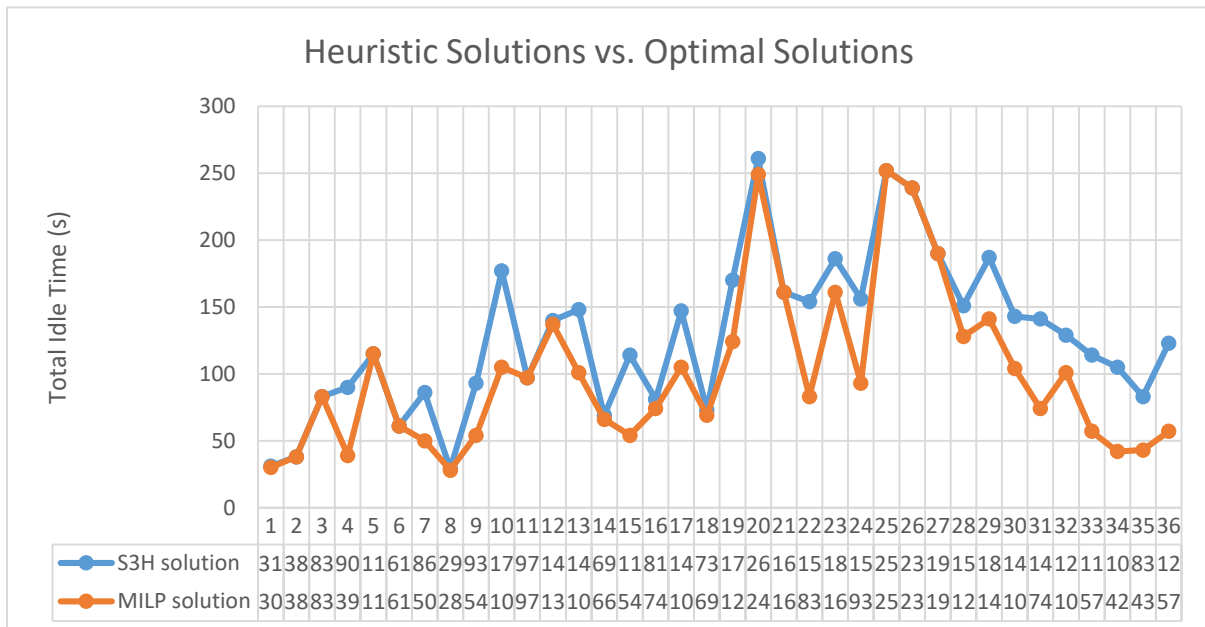


Figure 9.10: Heuristic Solution discrepancies in comparison to Optimal Solutions

The individual percentage discrepancies were calculated and synthesised. The results are summarised in Table 9.4.

Table 9.4: Discrepancy results synthesis

Average (%)	Median (%)	Minimum (%)	Maximum (%)
40.17	22.85	0	150

Figure 9.11 shows the discrepancy percentages between heuristic and optimal solutions when the problems were arranged according to increasing fixture quantity.

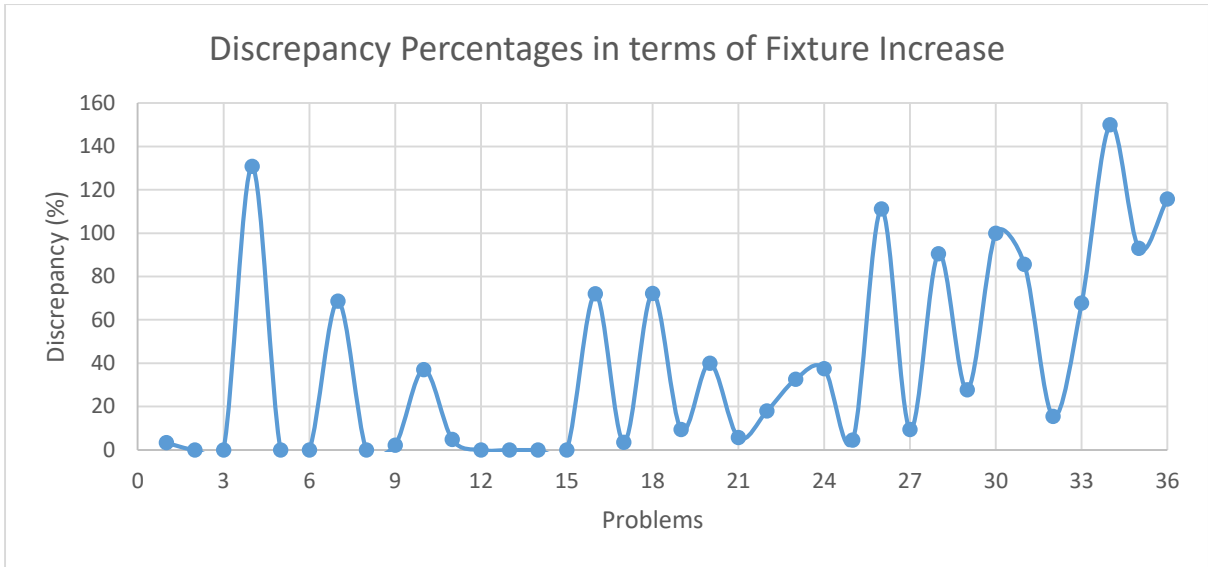


Figure 9.11: Discrepancy Increase in terms of Fixture Increase

Figure 9.12 shows the discrepancy percentages between heuristic and optimal solutions when the problems were arranged according to increasing part quantity.

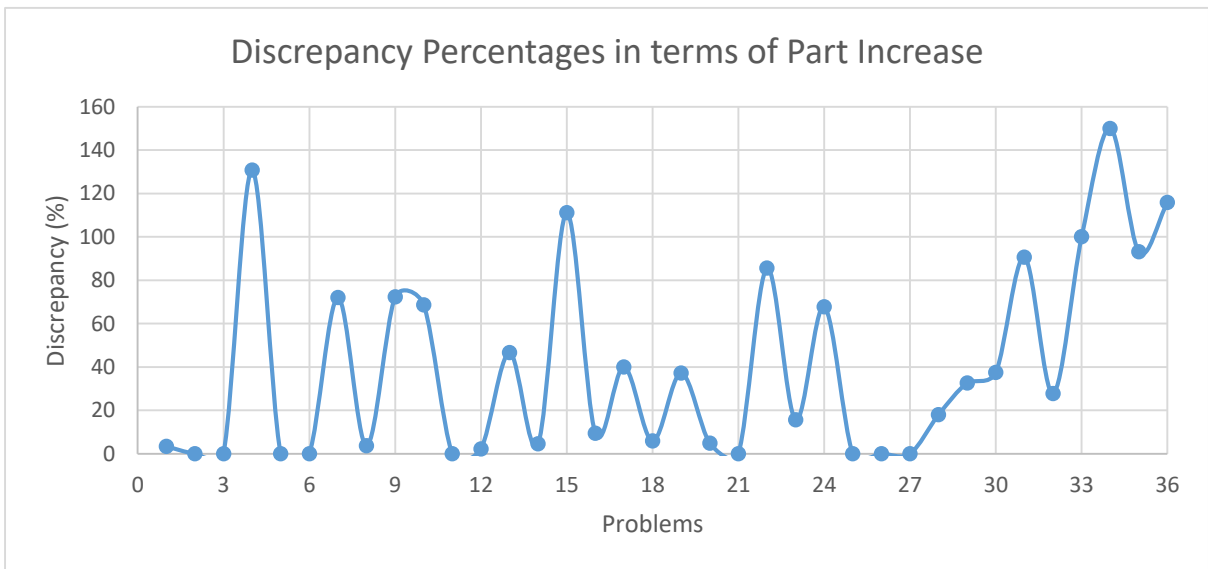


Figure 9.12: Discrepancy in terms of Part Increase

Figure 9.13 shows the comparison between the Stage III Heuristic solution times and those of the MILP model for the same test problems. The test problems are arranged in order of increasing variables (and thus, complexity) of the MILP model problem formulation.

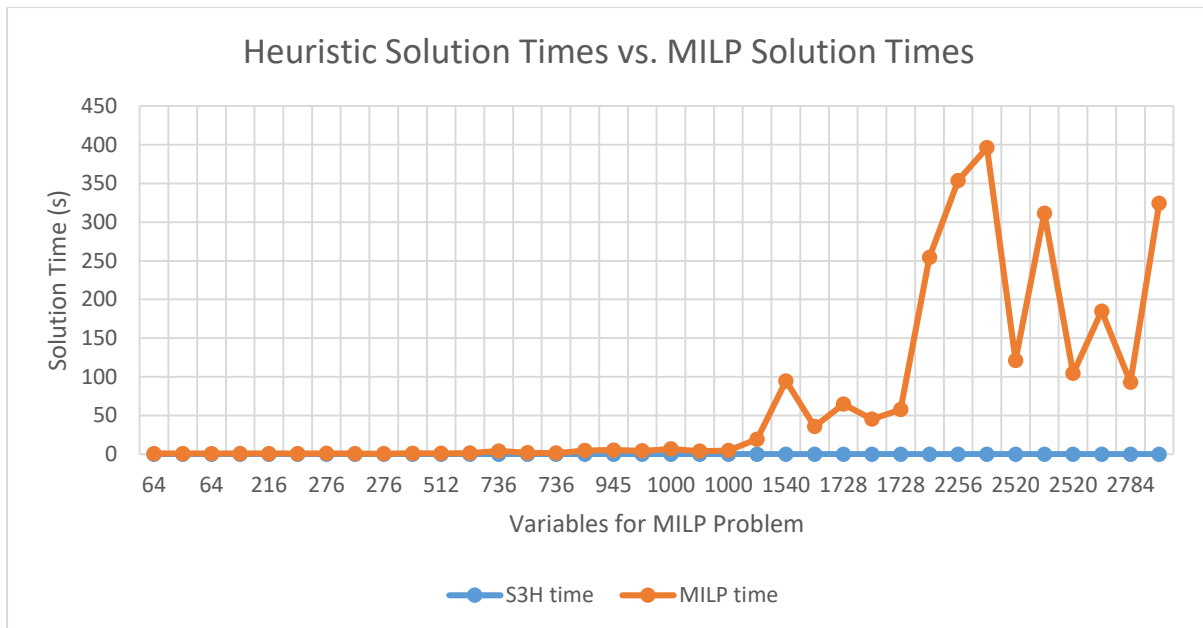


Figure 9.13: Heuristic Solution Times in comparison to MILP Solution Times

9.6.4 Analysis

The average heuristic solution was just above 40 % greater than that of the optimal solution. The median, however, was much lower at 22.85 %; this indicates that the majority of solutions were closer to the optimal solution than within 40 %. The maximum discrepancy was found to be 150 % greater than the optimal solution, generated for one of the 12-part/6-fixture problems (which were the most complex in terms of variables for the MILP model). The heuristic produced solutions that were identical to the optimal solution in nine instances, which made up 25 % of the sample set. The solutions only matched for 2-fixture problems. The observation can be explained by the relative lack of feasible solutions available to the solver in these cases, due to the combination of the following factors: the same fixture must not be scheduled in consecutive time periods, and; the intracluster order must be upheld. The solutions do not always coincide for 2-fixture problems, as shown by six of the 2-fixture problems tested, i.e. 40 % of such cases. However, it should be noted that the 2-fixture problems would have skewed the data in favour of the heuristic.

Figure 9.11 shows the behaviour of the discrepancy percentages when the problems were arranged according to increasing fixture quantity. Given the phenomenon that occurred for many 2-fixture problems, it was expected that the discrepancies may increase continuously; however, the graph revealed that the discrepancies fluctuated greatly, and no trend line could match the data points with an agreeable coefficient of determination.

Figure 9.12 shows the behaviour of the discrepancy percentages when the problems were arranged according to increasing part quantity. The discrepancies showed high fluctuation as for Figure 9.11, and an agreeable coefficient of determination was similarly elusive.

Figure 9.13 shows the superiority of the S3H in terms of solution times for the same problems. The solution time discrepancies remain marginal until the complexity of the MILP problem reaches 1000 variables, which is where a significant divergence emerges. The S3H solution time continues to remain

under 0.2 seconds, while the MILP solution reaches solution times close to 400 seconds for the sample set. As stated in Section 9.5, the MILP model is unable to reliably solve problems with greater than 12 fixtures. The S3H, however, was able to solve a 12000-part/600-fixture problem in 1.6135 seconds.

9.6.5 Conclusion

The heuristic performed reasonably well in comparison to the MILP model, with discrepancies in total idle time averaging 40.17 % for the sample set. The discrepancies showed no definite trend in relation to problem complexity, apart from the most complex problems producing some of the highest discrepancies. This was expected due to the greater availability of feasible solutions that the B&B solver can explore; as opposed to the greedy approach of the S3H, which is unable to backtrack to find better solutions.

Solution times for the heuristic were almost negligible; every problem was solved within 0.2 seconds. It can be concluded that the heuristic provides good, feasible (near-optimal) solutions with substantial savings in solution time.

9.7 Simulation Behaviour

9.7.1 Aim

To investigate the behavioural characteristics of the FxMC via the simulation, and by doing so, assess the effects of implementing the final schedule in an operational cell.

9.7.2 Methodology

The S3H was used to create datasets from which the agent-based simulation could run. Large datasets that were not feasible for the MILP model were implemented in the simulation via the heuristic. Problem sizes consisted of part quantities that increased from 100 parts to 400 parts in increments of 100 parts. The fixture quantities used for each part quantity was chosen such that the first sample assigned 20 parts to each fixture and the second sample assigned 10 parts to each fixture. The operation times were randomised within the 30 – 90 second range used in Section 9.5. The same seed was used for the MATLAB® random number generator, which resulted in time values being appended to the current list as part quantity increased; this resulted in less variation in the datasets. The tests were not repeated, as comparisons were to be made between the S3H results and the simulation results, and not between each other. As such, the eight samples evaluated were deemed sufficient.

- Test range: 100:100:400 parts; fixtures = 0.05*parts & fixtures = 0.1*parts
- Repetitions per test: 1
- Criteria: Idle Times, Makespan and Utilisation

9.7.3 Results

The results presented in Appendix A.6 were synthesised. The total idle time and makespan for both the simulation and Stage III Heuristic are summarised in Table 9.5.

Table 9.5: Simulation/Heuristic comparison results synthesis

Simulation		Stage III Heuristic		Total Idle Time Discrepancy		Makespan Discrepancy	
Total Idle Time (s)	Makespan (s)	Total Idle Time (s)	Makespan (s)	Absolute (s)	Percentage (%)	Absolute (s)	Percentage (%)
100 parts, 5 fixtures							
1479	7345	1108	6621	371	33.46	724	11.06
100 parts, 10 fixtures							
1183	7212	796	6471	387	48.59	741	11.45
200 parts, 10 fixtures							
1912	13979	1110	12505	802	72.24	1474	11.78
200 parts, 20 fixtures							
1612	13843	780	12340	832	106.67	1503	12.18
300 parts, 15 fixtures							
3346	21133	2151	18928	1195	55.74	2205	11.65
300 parts, 30 fixtures							
2486	20725	1206	18453	1280	106.14	2272	12.31
400 parts, 20 fixtures							
3651	27748	1977	24758	1674	84.68	2990	12.08
400 parts, 40 fixtures							
3334	27615	1511	24533	1823	120.65	3082	12.56

Figure 9.14 shows an overlay of the idle times that resulted from cellFR (where fixture reconfigurations occur) for both the simulation and heuristic for the 100-part/5-fixture problem.

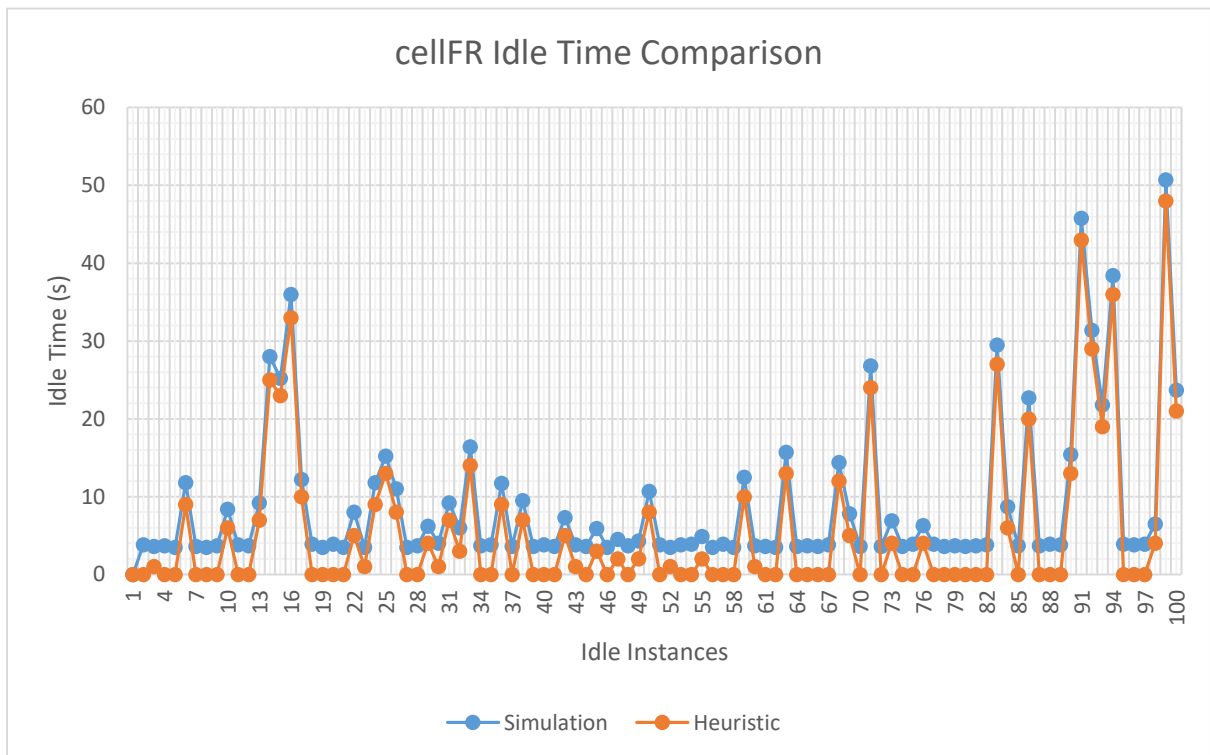


Figure 9.14: cellFR Individual Idle Time overlay (100 parts, 5 fixtures)

Figure 9.15 shows an overlay of the idle times that resulted from cellPP (where part processing occurs) for both the simulation and heuristic for the 100-part/5-fixture problem.

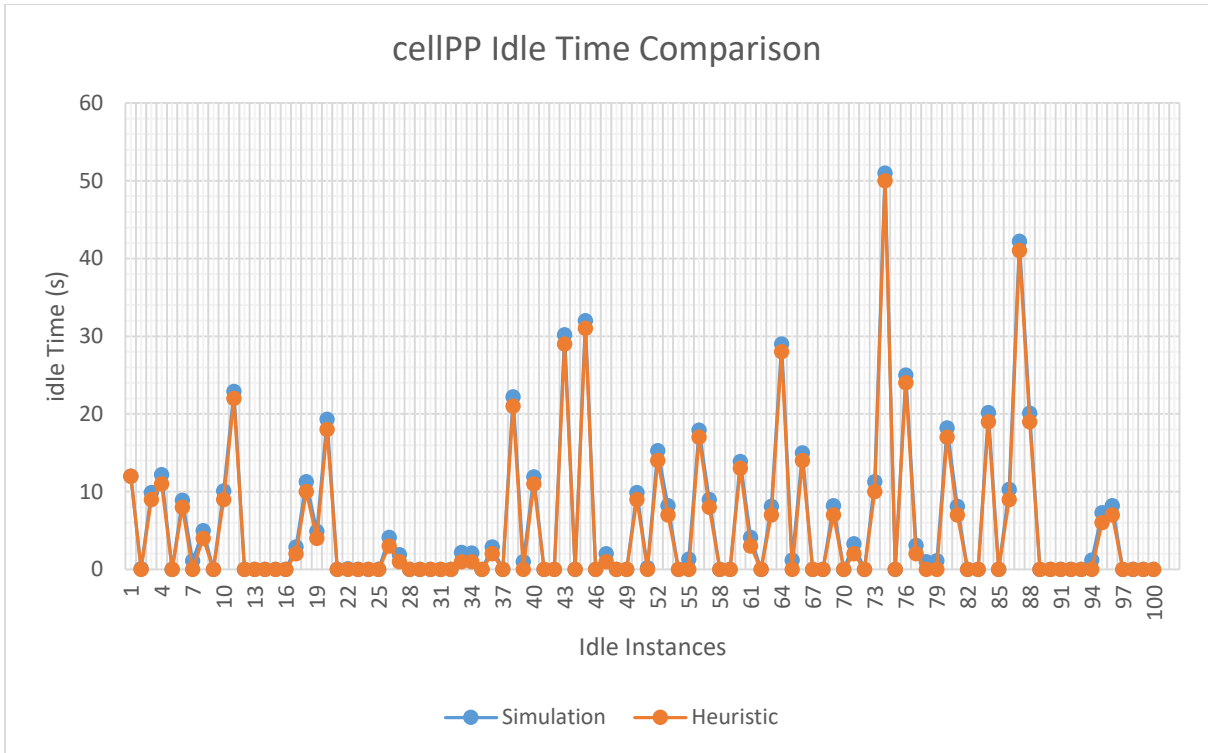


Figure 9.15: cellPP Individual Idle Time overlay (100 parts, 5 fixtures)

Figure 9.16 shows the idle time graph for cellFR generated from the simulation run for the 400-part/40-fixture problem.

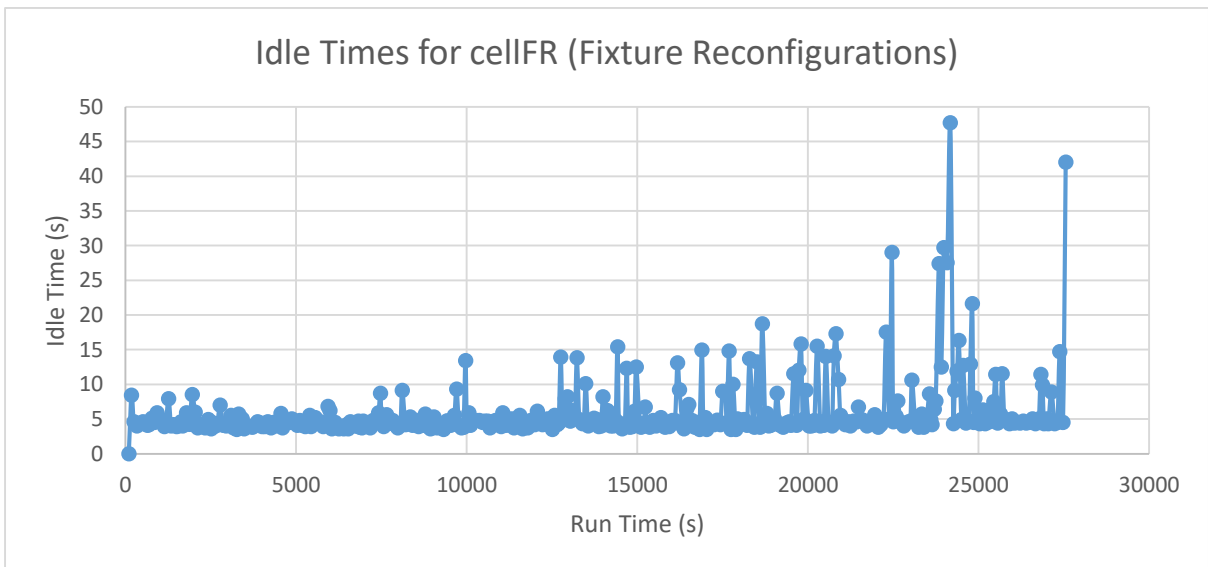


Figure 9.16: cellFR Individual Idle Times (400 parts, 40 fixtures)

Figure 9.17 shows the idle time graph for cellPP generated from the AnyLogic® simulation run for the 400-part/40-fixture problem.

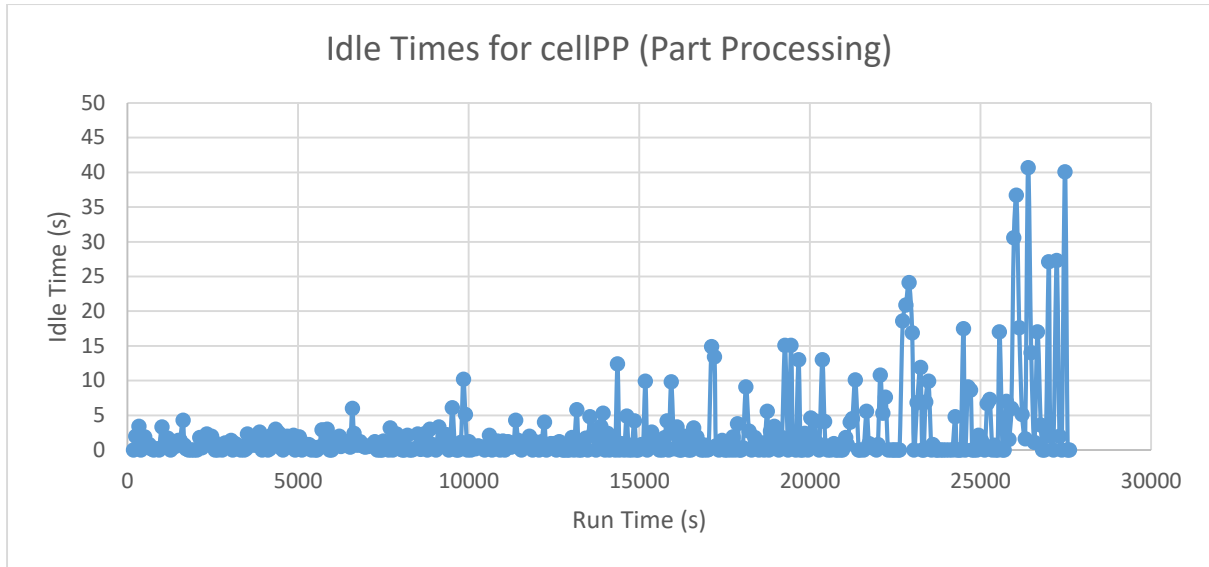


Figure 9.17: cellPP Individual Idle Times (400 parts, 40 fixtures)

Refer to Appendix A.6.1 for the other idle time graphs generated from the simulation for the problems that were investigated.

Table 9.6 shows the average utilisation for both cellFR and cell PP for the problems tested.

Table 9.6: Utilisation results synthesis

Average Utilisation	
cellFR	cellPP
100 parts, 5 fixtures	
0.8259	0.7684
100 parts, 10 fixtures	
0.8416	0.7757
200 parts, 10 fixtures	
0.8587	0.8383
200 parts, 20 fixtures	
0.8655	0.8347
300 parts, 15 fixtures	
0.8419	0.8505
300 parts, 30 fixtures	
0.8622	0.8541
400 parts, 20 fixtures	
0.8595	0.8554
400 parts, 40 fixtures	
0.8642	0.8561

Refer to Appendix A.6.2 for the utilisation graphs generated from the simulation for the problems that were investigated.

9.7.4 Analysis

Table 9.5 shows that the discrepancy percentages for total idle time increased for increasing fixtures; this was due to a relatively constant absolute discrepancy (in seconds) for similar part quantities, which resulted in higher percentages for greater fixture quantities due to the reduced total idle times that ensued. It was expected that the discrepancies would have emanated from a constant behaviour that produced the similar incongruity for the same number of parts.

The individual idle times for the simulation runs were recorded separately, depending on whether cellFR or cellPP was idle; this produced the graphs shown in blue in Figure 9.14 and Figure 9.15 for the simplest sample problem tested (100 parts, 5 fixtures). Figure 9.14 and Figure 9.15 include an overlay of the individual idle times extracted from the heuristic results (shown in orange).

Figure 9.14 shows a constant discrepancy whenever the heuristic data is zero; an idle time of around 4 seconds is shown to exist in the simulation for these cases. The reason for the inconsistency was that the heuristic only dealt with operation times, whereas the simulation also regarded transportation time. The simulation was developed such that fixtures were returned to fixture storage before the next fixture was dispatched; this was to ensure the availability of every fixture for the next operation (apart from the ineligible fixture that was reconfigured in the previous time period). The behaviour of the simulation meant that the reconfigured fixtures briefly waited in cellFR for the fixture from cellPP to return to storage; this resulted in an accumulation of additional idle time in cellFR for every instance of both cellFR and cellPP being idle.

Figure 9.15 displays superior consistency with the heuristic results in comparison to Figure 9.14. The reason for the consistency in Figure 9.15 is that the idle time is only counted while the fixture is in cellPP; cellPP releases its fixture immediately when not waiting for cellFR to complete its operation, thus avoiding additional idle times being accumulated above those of the heuristic.

Slight discrepancies apart from those already described were observed in both Figure 9.14 and Figure 9.15; these can largely be attributed to the transportation times associated with the agents in the simulation and the computational time required to select values from databases.

Figure 9.16 and Figure 9.17 display the idle time graphs generated for the most complex problem tested (400 parts, 40 fixtures). The problem was used to provide insight into the general behaviour of the heuristic solutions for large-sized problems. Larger-sized problems displayed much lower idle times for the majority of instances, due to the abundance of candidates available to the algorithm for selection. Idle times ≤ 5 seconds make up the majority of data points in Figure 9.16 (when disregarding the shift in results due to transportation time for cellFR) and Figure 9.17; whereas the majority of idle times in Figure 9.14 and Figure 9.15 are shown to fluctuate a lot more and exist within a larger range. A distinction is made at the later segments of Figure 9.16 and Figure 9.17, where the idle times are shown to increase severely. The increase is caused by the scarcity of candidates for the algorithm to select from as the solution nears its end. The increase is exasperated by the intermittent enforcement of the 'forced computation' (see Section 6.10) at this stage of the solution, due to the depletion of columns that continuously produced favourable candidates, thus forcing the algorithm to ignore some of the subsequent best candidates in favour of generating a feasible solution. The trend of abrupt increases in idle times towards the end of the solution was observed in every case tested apart from the 300-part/15-fixture problem (as seen in Appendix A.6.1). The cause of the deviation was unclear; it is believed that the results transpired in that manner due to the random selection of the initial candidate combined with the time values used for the operation times, leading to the solution shown; high idle times are shown very early in the graph. It can be concluded that a high increase in idle times towards the end of the solution should be expected, but the phenomenon is not guaranteed.

Utilisation of cellFR and cellPP were also observed for this test, the results for which are displayed in Table 9.6. The utilisation graphs in Appendix A.6.2 reveal a very similar trend for every test. Utilisation of cellFR and cellPP begin at a low value due to the initial ramping up of the simulation (creating and storing fixtures). The utilisation of cellPP is additionally affected by the delay of having to wait for cellFR to complete its first operation. The utilisation values stabilised at around 0.85 for every test. The shorter run times for the 100-part problems resulted in lower average utilisation values, due to the ramping up phase constituting a larger proportion of the makespan. The average utilisation of both cellFR and cellPP remained within a close range to 0.85 for the larger problems tested. The range compares favourably to the 0.7 – 0.9 utilisation range suggested by Liao [38] for cellular manufacturing systems. The high levels of utilisation are facilitated by the attempts to minimise idle time through the S3H.

The results from the test sample verify the feasibility of the solutions produced by the Stage III Heuristic, in that the solution was generated and tested to completion via the simulation. The verification can be used to conclude that the MILP model was in-fact limited by the computational expense of the solver, not the model itself; feasible solutions for part quantities greater than 12 do exist, but were not found by the B&B algorithm within reasonable time.

9.7.5 Conclusion

The results revealed the effect of transportation time on the operation of the fixture manufacturing cell. A constant accumulation of idle time in cellFR was observed due to the delay associated with returning the fixture used for part processing to fixture storage.

The results for the simulation and heuristic corresponded closely, apart from the shift in the cellFR graph (Figure 9.14) due to the problem described. Minor discrepancies were observed due to transportation time and computation time.

The utilisation of cellFR and cellPP stabilised at around 0.85 for every test, which compared favourably to literature. These results showed that the minimisation of total idle time yielded a highly utilised, and thus efficient, cellular manufacturing system.

The simulation results verified the feasibility of the heuristic solutions. The feasibility of the heuristic solutions proved that feasible schedules do exist for larger-sized problems. This confirms that the limitations experienced by the branch and bound solver for the MILP model were due to the computational expense of finding the optimal solution, instead of the non-existence of a feasible solution.

9.8 Summary

Tests were conducted on the main aspects of the research undertaken. The individual sections detail the findings observed and the conclusions drawn from those results. Key observations include:

- The MDS procedure was robust for larger-sized problems, but the effect on accuracy was unavoidable.
- The dissimilarity measure and k-means clustering technique yielded good clusters, verifying the suitability of their application here.
- The optimal leaf order algorithm consistently produced the optimal intracluster sequence.
- MILP model problems can be reliably solved for problem sizes up to 12 parts, with a 100% success rate in obtaining the optimal solution via the B&B algorithm.
- The Stage III Heuristic produced near-optimal; but the likelihood of high quality solutions decreases for larger-sized problems due to the greedy approach.
- The agent-based simulation revealed the effect of transportation time on the behaviour of the fixture manufacturing cell.
- High levels of utilisation were observed, as should be the case for a system with minimised idle time.
- The S3H results were verified as feasible through the simulation results for large-sized problems; this confirmed the technical limitations of the solver for the MILP model (as opposed to a formulation error).

10 Discussion

10.1 Introduction

This chapter presents a summary of the dissertation, with elaboration on the research findings, performance and contributions. Research shortcomings and recommendations for future work are also discussed. The fixture manufacturing cell is then discussed in context of its implications in practice.

10.2 Concept Overview and Justification

A survey of current literature was conducted in Chapter 2. The research aimed to present the on-demand Fixture Manufacturing Cell (FxMC) as a solution to the management of reconfigurable fixtures in a mass customisation production system. The studies reviewed revealed an absence of research regarding the scheduling of reconfigurable fixtures. There is a prevalence of research on the optimisation of the fixture design procedure [6], [23], [26] while studies on the scheduling thereafter are lacking. The scheduling of standard fixtures was found to be scarce, with few relevant papers discovered [7]–[10]. The studies mainly handled fixtures through a single constraint that limited the availability of the resource in the model. Bi et al. [12] reiterated the observation made by Bi and Zhang [4] that utilising modular fixture components efficiently in production planning was yet to be explored. The literature reviewed revealed that this research gap remains, despite the increased interest in mass customisation, for which reconfigurable fixtures is a major facilitator [2].

Mass Customisation (MC) is a concept that relies on flexibility and responsiveness to adapt to ever-changing customer demands, while maintaining cost-effectiveness and high quality standards [13]. MC can be interpreted as the maximisation of the advantages of both low volume and high volume production, whilst minimising their respective disadvantages. The research undertaken aimed to investigate the on-demand fixture manufacturing cell as a facilitator of MC. The decoupling point of the test product was definable as ‘mass customisation’ by the definitions of both Squire et al. [16] and Tien et al. [17], due to the theoretical handling of the test product as described throughout Chapter 5. The modularity of manufacturing systems [2], [20] and fixtures [4] were found to be feasible approaches for the implementation of MC in practice. The research undertaken made use of modularity for both the cellular manufacturing systems selected for the production system (Section 3.5) and the fixture design (Section 3.4).

Reconfigurable fixtures was noted as a major facilitator of mass customisation. The existing fixture types were reviewed. The fixture design was not a primary research objective, but a platform upon which the concept could be investigated. It was decided that the research would implement a modular fixture, due to its popularity in industry and ease of implementation [4]. The design implemented was that of a grid hole base plate, with dowel pin modules. The base plate array consisted of 8×8 holes. The pin range was limited to 8 – 16 pins per configuration. The design restricted the customisability of the fixture, but it was determined that a total of 7.1325×10^{14} different pin configurations could be assembled within these constraints. An engraved plaque was selected as the theoretical test product. Customisability of both the border shapes for pin configurations, and engravings for part processing

times were possible through the test product; this ensured sufficient differentiation in the jobs assigned to the scheduling method. The engraving process was found to produce a sufficiently low thrust force on the part, such that the fixture design rigidity was deemed adequate for the application.

Manufacturing systems that could facilitate mass customisation were investigated. Reconfigurable Manufacturing Systems (RMS) were of considerable interest. RMS offered a compromise between the flexibility of flexible manufacturing systems and the cost-effectiveness of dedicated manufacturing systems. The characteristics of RMS, as described by Koren and Shpitalni [33], align with the flexibility and cost-effectiveness that mass customisation strives to achieve. The fixture manufacturing cell was suggested to provide process and product flexibility through the circulation of reconfigurable fixtures in the manufacturing system; and volume, expansion and product flexibility dependant on the design of the specific cell. The concept of the fixture manufacturing cell is aligned to the six characteristics of RMS; the concept of centralising the fixture fabrication and management in a specialised cell on the shop floor can provide the customisation, convertibility, scalability, modularity, integrability and diagnosability that is required in a RMS.

The concepts of Group Technology (GT) and Cellular Manufacturing (CM) systems were investigated. CM systems were found to be practicable for the implementation of RMS, due to the improvements in efficiency and adaptability of the specialised manufacturing cells. GT was also found to be advantageous for the application of modular fixtures, due to the degree of customisation being narrowed to within the domain of the part families that the cells are specialised for. The characteristics of CM systems justified the structure of the two-cell production system investigated in the research as a representation of a mass customisation system (Section 3.6).

10.3 The FxMC

The FxMC layout was formulated as a semi-automated, single-operator cell (Section 3.5). The cell components were separated into: fixture storage, fixture raw material inventory, fixture fabrication station, fixture reconfiguration station, control station, and transportation system. The layout was such that unidirectional flow was utilised, and operator movement was unobstructed. The cell was designed such that fixtures could be either: delivered as-is to the part, if already in the required configuration; or directed to the fixture reconfiguration station, where it is reconfigured before being dispatched to the part. The transportation system layout was such that fixtures could be readily recirculated in the production system. The operator was located near the control station and capable of moving from fixture fabrication station to the fixture reconfiguration station at ease. Despite the scope for increased automation, the structure and function of the semi-automated, single-operator cell described remains valid for industrial implementation. Improvements in efficiency could be obtained with a higher degree of automation, at the expense of time and cost of both implementation and maintenance.

The FxMC layout was used as a basis for the proof-of-concept of the on-demand fixture manufacturing cell, i.e. the Lab FxMC (Chapter 4). The implementation of the Lab FxMC necessitated the integration of mechanical, electrical and electronic hardware, together with programmable software, to assemble and automate the cell. The Lab FxMC was primarily comprised of: ASRS for fixture storage; control station with PC and PLC; workstation for fixture reconfigurations; CNC router for fixture fabrication; automated conveyor system for transportation; and pneumatic actuator for route flexibility.

Modifications were made to enhance the performance of the existing equipment; this included the hardware and software of the ASRS and CNC router. RFID tags were embedded into the pallets that transported the fixtures. The conveyor system was automated through a soldered relay circuit and PLC unit. A pneumatic circuit was implemented to divert fixtures to the reconfiguration station if required. The PLC automation extended to the pneumatic circuit, where a sensor was used as an input to activate the route diversion. Mounting components were 3D-printed from PLA plastic. A ladder logic programme was developed for the PLC functions, which automated the circulation of fixtures through the Lab FxMC.

The CNC router was used to fabricate the test fixtures. Rough testing yielded the estimated operation time ranges of 30 – 90 seconds, and the transportation times for the agent-based simulation. The throughput for the ASRS was found to be considerably slow; an average of 88 seconds was determined for both storage and retrieval activities. The considerable transportation times in the Lab FxMC, and the impracticality of conducting up to 400 reconfigurations for a single test, prompted the requirement of building a simulation for testing. The ASRS transportation times used in the simulation were closer to industry observations.

The Lab FxMC served to provide a practical demonstration of the FxMC concept. The layout, unidirectional workflow, and operator convenience of the Lab FxMC was as intended from the FxMC layout. These fundamentals would also remain true for industrial implementation, with the expectation of increased automation and scale. The theoretical formulation of the FxMC layout (Section 3.5) together with the practical execution of the functioning Lab FxMC (Chapter 4) achieved Objectives 1 – 3 of the research (Section 1.4).

10.4 PPC System with FxMC

Production planning techniques were reviewed (Section 2.6). The Manufacturing Resource Planning (MRP II) system utilised the push production technique, which results in high Work in Process (WIP) and inventory. Just-in-Time (JIT) systems utilise the pull production technique, which minimises WIP and inventory. It was decided that the Production Planning and Control (PPC) system of the research implementation (Chapter 5) would be based on the push production technique of MRP II.

The PPC system was responsible for deriving the data required by the scheduling method from the customer order. MRP II was used to produce and update the Product Data Management (PDM), Master Production Schedule (MPS) and Bill of Materials (BOM) required for the products. The main function of the push technique was to ensure high levels of inventory (based on product forecasts). As stated in Section 5.2, Walker and Bright [97] determined that statistical behaviour of customer decisions in a mass customisation environment was only stabilised at high levels of WIP. The unpredictability of consumer behaviour means that companies operate at a high risk if inventory levels are low, which could be detrimental to the manufacturer in terms of product delivery and customer satisfaction. Therefore, the high levels of inventory resulting from MRP II and the Reorder Point (ROP) system was favoured for the PPC system implemented. The PPC system provided a framework within which the scheduling method would operate. As such, MRP II was also conducive to the deterministic data required for the scheduling method, whereby the PDM, MPS and BOM data should be known prior to initiating production of the order (which was dealt with in batches by the scheduling method). This was

a limitation due to the novel problem structure discouraging a stochastic method being formulated as an initial undertaking of a scheduling method for reconfigurable fixtures. MRP II was also favourable for the test product implemented (Section 3.3), whereby high levels of inventory could be maintained as the customisation of the product was limited to a similar raw material inventory (two-dimensional plaques of same material only). Thus, inventory orders for these products can be placed without significant risk of the material not being used due to variations in customer orders.

The production system representation of the research (Section 3.5), however, operated as a JIT system. The lack of buffering before the part processing cell resulted in low WIP (pull production) for the tasks in which reconfigurable fixtures were required, and the unit workflow was akin to the workflow characteristics of the Kanban system. This ensured that the use of fixture inventory was kept lean so that fixture utilisation was optimised. The pull techniques used in the operation of the shop floor would yield the benefits of lean manufacturing; these include minimal wastage of resources caused by producing and holding a high quantity of fixtures and improved control due to unit workflow.

The result of the amalgamation of push and pull concepts is: a mass customisation system that operates as a push production system in terms of customer orders, inventory and broader management; but a cellular manufacturing system on the shop floor, which operates as a pull production system due to minimal buffering and unit workflow.

A PPC system was described in Chapter 5, where the activities of the broader production system incorporating the fixture manufacturing cell for mass customisation was elaborated. The system would rely on the capabilities of an advanced CAD/CAM software to provide the parameters from which the scheduling can be formulated, such as fixture configurations and operation times based on the customised product. The reorder point system was chosen for inventory management, which is evident in push production systems. This leads to higher inventory levels, but reliable product delivery in an unpredictable industry. The product data management, master production schedule and bill of materials were elaborated on; where the MPS and BOM were separated to account for the different requirements of fixture management and product management. The scheduling method of the research was described as being integrated into the shop floor control of the MRP II system. The scheduling method would utilise the information provided to it from PDM (for pin configurations, projected operation times, and other such parameters) and the MPS (for initial scheduling data of fixture and part requirements); the BOM would provide management information for inventory control and fixture fabrication (beyond the scope of the research objectives). It was determined that, based on the recirculation of fixture resources in the system, the MPS and BOM would have to be separated for parts and fixtures. The MPS is only determinable until the scheduling results are obtained. The BOM was found to be indeterminable until after the scheduling method if fixture fabrication is considered. Thus, the PPC system provided a framework for both integration of the fixture manufacturing cell, and information flow from the customer order (as per mass customisation decoupling point) to shop floor control (where the scheduling method is incorporated).

10.5 Scheduling Method

A scheduling method for the optimal management of reconfigurable fixtures through an on-demand fixture manufacturing cell for mass customisation production systems was developed. The method was

separated into three stages: Clustering (Stage I), Intracluster Sequencing (Stage II), and Final Scheduling (Stage III). The formulation (Chapter 6) and validation (Chapters 8 and 9) of this method (and the alternative heuristic) achieved Objective 4 and Objective 6 of the research (Section 1.4); an optimisation model was developed, which was shown to minimise the total idle time for the problems tested.

The method was based on the observation that: minimising the reconfiguration effort per fixture for successively reconfigured pin configurations would reduce individual fixture reconfiguration times; and synchronising the fixture reconfiguration and part processing operations scheduled per time period would minimise total idle time.

The Analysis sections in Chapter 9 are recommended for detailed examination of the results discussed hereof, as only a summary of those findings are presented in this chapter.

10.5.1 Stage I

Stage I (Section 6.7) dealt with the initial grouping of parts to fixtures. The goal of this stage was to ensure that similar part shapes (and thus pin configurations) were assigned to the same fixture base plate. The procedure involved a dissimilarity measure, Multidimensional Scaling (MDS) and k-means clustering.

The binary dissimilarity measure (Equation (6.3)) was based on the Rand index for similarity (or simple matching coefficient). Binary representation of the data was chosen for computational efficiency. Equation (6.3) was structured such that mismatches were significantly penalised through the exponential weightings of 2. The weightings were justified by the increase in reconfiguration effort that resulted from mismatches; with pin manipulations (removals/insertions) warranting a time penalty increase for the task. This resulted in a close correlation between the dissimilarity measure and the practical execution of the task it represents. The structure of Equation (6.3) was such that similar pin configurations yielded a lower value (higher dissimilarity); this was so that the measure could correspond to a distance, whereby shorter distances were favoured by the clustering algorithm.

The dissimilarity measure yielded a non-metric distance matrix that could not be clustered directly; as such, MDS was used to scale the higher-dimensional data to a two-dimensional map. The resultant stress value from the MDS procedure indicated the usability of the scaled data. An evaluation table [105] was used to gauge the probability of the data being without structure. Section 9.2 revealed that scaled data from higher dimensions (≥ 20 parts) consistently yielded stress values of around 15 % below their respective threshold values. The results for lower dimensions (10 parts) were less robust, exceeding the threshold value on several occasions. It was concluded that scaling from lower dimensions should be monitored on a case-by-case basis to ensure that the scaled data is within the specified stress value. However, as the emphasis of the research was on mass customisation, it can be expected that most job lists would consist of higher part quantities, thus eradicating the risk of unstructured data. Despite the satisfactory results for higher part quantities; the stress values were consistently above 0.1, meaning that an inherent loss of accuracy is unavoidable with the MDS procedure. This means that the 2D map does not represent the non-metric distance matrix with 100 %

accuracy; however, this non-ideality was expected due to the non-Euclidean properties of the original data.

The k-means clustering algorithm was used to group similar parts to be assigned to the same fixture base plate, based on the closeness of their data points on the 2D map. The number of clusters was predetermined, based on the number of fixtures available. It was realised that using mean values as cluster centres was more appropriate to the application than using median values (such as for k-medoids, density-based scan, or p-median methods [102]). The purpose of using median values is based on the elimination of outliers; however, all parts represented on the 2D map have to be assigned to a fixture in the research application. The mean values of the k-means algorithm ensured that a closeness of all cluster objects to the cluster centre could be achieved. The goodness of the clusters formed were evaluated by the silhouette values generated. Section 9.3 details the testing of clustering 100 parts to various quantities of fixtures; a total of 30 such tests were conducted. The average silhouette values were around 0.5 (for a range of [-1; 1]); 1.67 % of the objects yielded silhouette values below 0, the lowest of which was -0.2839; it was concluded that clear, unambiguous clusters were generated for the tests. This further validated Equation (6.3) as an appropriate dissimilarity measure for the application. Other observations from the tests were made. The goodness of clusters are independent of the number of fixtures clustered to. A heuristic (Figure 6.9) was required to deal with high part quantities being clustered to two fixtures so that a feasible final schedule could be obtained in Stage III; this affected the goodness of the clusters formed thereof. The heuristic was only required for clustering to 2 fixtures for 100 parts. The clustering technique means that unevenly-sized clusters are often formed. Evenly-sized clusters were considered, but this would significantly affect the goodness of the clusters formed, with the only uptake being a more uniform utilisation of the fixture base plate; as such, this endeavour was regarded as insignificant.

The output of Stage I was clearly defined part families that could be sequenced within that group for that fixture.

10.5.2 Stage II

Stage II (Section 6.8) dealt with sequencing the groups formed in Stage I. Hierarchical clustering was used with single linkage to construct a dendrogram through an agglomerative procedure. The single linkage ensured that subtrees of the dendrogram were formed on a basis of closest objects. The distances used for this stage were retrieved from the non-metric distance matrix for the true pairwise distances. The optimal leaf order algorithm was then used to obtain the sequence for which the total cumulative pairwise dissimilarity was minimised. The procedure is to be repeated for each cluster formed in Stage I. Section 9.4 revealed that the reordering of the initial (arbitrary) order of the clusters formed in Stage I consistently yielded improvements. The average improvement per cluster ranged from 5.85 % to 14.34 %, except for instances where node-flipping was not possible, i.e. two objects in cluster, or initial order the same as optimal. The minimisation of total cumulative pairwise dissimilarity per fixture is akin to minimising the distance traversed along that fixture; it ensures that the order in which reconfigurations on that fixture are made minimises the pin manipulations; this completed the initial objective of minimising reconfiguration effort per fixture for successively reconfigured pin configurations. It should be noted that the total cumulative pairwise dissimilarity is true for both the

forward and reverse order of the intracluster sequence, as the non-metric distance matrix is a symmetric matrix.

10.5.3 Stage III

Stage III (Section 6.9) synchronises the fixture reconfiguration operations and part processing operations such that the total idle time is minimised. The solution was achieved through the formulation of a Mixed Integer Linear Programming (MILP) model. The MILP model utilised the input from Stage II, which specified the intracluster sequence, together with various other constraints and bounds to ensure a feasible solution could be found. The model was structured to represent the production system described in Section 3.5. The objective function minimised the absolute time differences for the operations scheduled in every time period. The constraints ensured that: the fixture-part mappings flow from the first operation to the second operation in successive time periods; the number of time periods corresponds to the number of synchronous operations; the intracluster sequence is upheld; each fixture-part mapping is assigned to a time period only once; each time period is assigned to a fixture-part mapping only once; and bounds enforcing the non-negativity and integer conditions for their respective variables. The first two constraints had to be expanded to linearise those constraints for the formulation of the MILP model (Section 6.9.2.1).

The problem was chosen to be formulated as a MILP model and solved with the exact solution technique of Branch and Bound (B&B). Exact methods are computationally intensive; they are not regarded as an efficient technique for solving the NP-hard structure of typical scheduling problems in manufacturing. However, an exact method formulation of a novel problem is regarded as a crucial initial step to understanding the structure of the problem, from which the development of effective heuristics and metaheuristics can be based and benchmarked [73]. Given the research gap regarding the research undertaken, it was decided that an exact method formulation would be of fundamental importance for the aforementioned reasons. Furthermore, the MILP formulation was selected over metaheuristics, such as genetic algorithm, since local search methods have a tendency to not escape local optima [66]; this was expected to be a concern for the problem structure, due to the constraints and variable sets described in Equations (6.5) to (6.14); the search space is very limited and specific, which could make finding feasible solutions problematic for metaheuristics.

Section 9.5 revealed numerous characteristics of the MILP model. Problems sizes within a range of 12 parts were tested. A 100 % success rate of solution convergence was achieved for the 75 problems. The constraint violations for three of the problems tested were negligibly small. The relative and absolute gaps were within 0.5, which agreed with the integer variables and parameters used. An average of 1.867 feasible solutions per problem was found. The growth characteristics of the solutions were observed. The number of variables (indicative of problem complexity) increased logarithmically for increasing fixtures, the rate of which increased slightly for larger part quantities. The number of variables increased polynomially for increasing parts, the rate of which increased slightly for larger fixture quantities. The polynomial increase was approximately to the third degree. The number of nodes explored corresponded linearly with the solution time, which proved to be the only reliable predictor of solution time. However, the general exponential increase in both nodes and solution time for increasing variables provided vital insight into the limitations of the MILP model.

Problems sizes consisting of more than 12 parts could not be reliably solved for the MILP model. The polynomial increase in variables for increasing parts, together with the exponential increase in both nodes explored and solution time for increasing variables, resulted in a computational expense that could not be reliably overcome by the solver. Testing in Section 9.7 revealed that feasible solutions for very large problem sizes do exist, despite the MILP solver being unable to obtain them; this observation vindicated the aforementioned reasoning for the MILP model limitations. This outcome was an expected drawback of the exact method used. However, the MILP model provided a fundamental understanding of the problem structure, and a benchmark on which the heuristic was developed and compared.

10.5.4 Stage III Alternative

The Stage III Heuristic (S3H) (Section 6.10) was developed to produce near-optimal solutions for Stage III in minimal time, and prove the feasibility of larger-sized schedules. The S3H is a greedy algorithm, which selects the lowest feasible idle time for each successive time period. The S3H used matrix manipulations to execute its operations, which produced an efficient algorithm in the MATLAB® software. The S3H included a ‘forced computation’, which ensured that the greedy approach did not eliminate the possibility of a feasible schedule being obtained.

Section 9.6 compared the performance of the S3H to the MILP solutions for a selection of 36 problems from Section 9.5. The S3H solution quality compared favourably to the optimal solutions. However, it was noted that the results were skewed in favour of the S3H due to the lack of variability in feasible solutions for the 2-fixture problems. The average discrepancy was found to be 40.17 % above the optimal solutions, with a maximum of 150 % observed. Discrepancy trends were investigated to no avail, as the discrepancy of the S3H solution fluctuated for both increasing fixtures and increasing parts. However, an uptake in discrepancy was noted for the most complex problems tested (12-part/6-fixture), one of which yielded the greatest percentage.

The solution times for the S3H and MILP model were compared. The S3H was found to be significantly superior to the MILP model in terms of solution times for the same problems; the most extreme cases yielded MILP times of order of magnitude 10^4 greater than the equivalent S3H time. The computational efficiency of the S3H was illustrated by a 12000-part/600-fixture problem being solved in 1.6135 seconds.

The Stage III Heuristic produced feasible solutions, as verified by the simulation tests in Section 9.7. The simulation runs were monitored and the behaviour of the schedules tested were as specified by the schedules. Larger-sized problems (up to 400 parts, 40 fixtures) were simulated, and the trends of individual idle times were observed. The idle time graphs revealed the effect of the greedy approach, which yielded very low idle times in the initial segments of the simulation runtime, but later increased considerably. The depletion of available candidates and the frequency of the ‘forced computation’, resulted in the algorithm having to select very high idle times. The inability of a greedy algorithm to backtrack (like B&B can) meant that the algorithm had to endure the options available by that point, to the detriment of the total idle time. The equivalent graph comparison for the sample problem MILP solution in Section 8.7, showed how the optimal solution sacrificed lower initial idle times in favour of producing the lower total idle time.

The S3H aimed to produce near-optimal, feasible solutions. The goodness of the solution quality was satisfactory, but increasingly inferior to the optimal solutions for the larger-sized MILP problems. It was recognised that finding feasible solutions for such highly constrained problems was likely to result in a deficiency of available solutions. Thus, the consistent obtainment of feasible solutions from the S3H warranted its efficacy.

10.6 Simulation

An agent-based simulation was created in AnyLogic® to simulate the behaviour of the fixture manufacturing cell and the part processing cell. The simulation was modelled through a process flowchart (Figure 7.6) with additional Java® commands to execute the desired behaviour thereafter. The simulation was required to verify the feasibility of the larger-sized schedules generated from the S3H, and monitor the real-time behaviour of the cell with relevant manufacturing performance metrics. The development (Chapter 7) and implementation (Chapters 8 and 9) of this simulation achieved Objective 5 and Objective 6 of the research (Section 1.4); the simulation resembled the Lab FxMC and was used for testing of the scheduling method, from which observations were made and conclusions drawn on the behaviour of the FxMC and its scheduling method.

Testing in Section 9.7 revealed the effect of transportation time on the results. The requirement of the fixture returning to storage after being used to secure the part ensures that the newly reconfigured fixture remains idle for longer than expected; this was shown to be ~ 4 seconds in the tests conducted. The results showed that transportation time has an influence in this case, but as it was a constant shift in the results, the observation was inconsequential to the final schedule produced from Stage III. The total idle time and makespan for the simulation results in comparisons to the S3H results, however, did show the constant discrepancy.

The total idle time values consistently decreased when fixture quantity was doubled for a constant number of parts. The improvements evident from Table 9.5 ranged from 9.51 % (400 parts) to 34.59 % (300 parts). The improvements were not independent of the number of fixtures, as the initial pivot fixture in the S3H was randomised and would have influenced the resultant idle times thereof. However, the results did reveal the improvement in total idle time, and thus makespan, for higher fixture quantities; this was due to the S3H being presented with more options from which higher quality solutions could be obtained.

The result revealed one of the advantages of having a higher fixture inventory, as it is conducive to obtaining improved schedules. For the scheduling method, a higher fixture inventory results in an increase in valid options from which the optimum is selected. For Stage I, an increase in fixtures ensures pin configurations that are less dissimilar (requiring fewer pin manipulations) are clustered together, yielding superior clusters with higher silhouette values. Conversely, a lower fixture inventory increases the possibility of outliers being grouped with parts that are only the least dissimilar of the available options (but requiring a high number of pin manipulations). The improved goodness of clusters in Stage I should improve the minimum cumulative pairwise dissimilarity on each fixture in Stage II, thus decreasing reconfiguration time per fixture. Stage III of the scheduling method is constrained to construct schedules where the same fixture must not be used in consecutive time periods (for both the MILP model and the heuristic). A high fixture inventory relaxes the severity of this constraint on the

result, such that the solver is more likely to find a solution yielding a lower minimum total idle time (objective function). The improvement in Stage III would be in conjunction with the lower reconfiguration times obtained by the intracluster sequences generated in Stage II, increasing the likelihood of an improved final schedule. A higher fixture inventory would also minimise the risk of fixture starvation due to damaged fixtures being discarded. This reduced consequence of discarding fixtures could also improve product quality, since the criteria for suitable fixture condition can be made more stringent, encouraging the disposal or repurposing of damaged fixtures.

The disadvantages of high fixture inventory are predominantly related to cost. An increased number of fixtures would result in resource expenditure associated with the operation of the fixture manufacturing cell to fabricate new fixtures. These would include energy consumption, labour costs, material costs and time. A higher fixture inventory would also require increased storage capacity for fixtures, and the holding costs thereof. These would include purchase, operation and maintenance of a larger storage device (such as an ASRS). Furthermore, low utilisation of fixtures would result from a comparatively high number of fixtures to jobs. Low utilisation decreases the value of each fixture to the manufacturing system, which is detrimental to achieving lean manufacturing goals. The scheduling method and production system in the research was designed to optimise fixture utilisation without compromising the advantages of an adequate fixture inventory.

The simulation also revealed the utilisation of the fixture manufacturing cell and part processing cell. The reduced total idle times of the S3H schedules resulted in average utilisation values of around 0.85 for the fixture manufacturing cell; the part processing cell, however, was affected by the additional idle time waiting for the fixture manufacturing cell to reconfigure its first fixture. The utilisation values were within the suggested range of 0.7 – 0.9, which renders the cell utilisation in the system as sufficiently high [38].

10.7 Shortcomings and Scope for Improvement

The research outcomes, while having satisfied the objectives outlined in Section 1.4, were met with deviations from ideal solutions that could provide scope for improvement in future research endeavours.

The non-zero stress values from the MDS procedure was unavoidable, but noteworthy. The clusters were formed from data that did not accurately represent the original non-metric distance matrix; the result of which is maintained through the successive steps. The MDS procedure was required for the k-means clustering algorithm to be used, such that clear and unambiguous clusters could be formed (unlike for hierarchical methods).

The scheduling method was limited by the assumptions presented in Section 6.6. The manufacturing system was assumed to be deterministic; this is rarely the case in practice, where robust stochastic models are necessary to adequately represent manufacturing systems [54]. However, as stated for the justification of the exact method in Section 2.7.2, the deterministic model developed for the research is a fundamental first step towards understanding the problem structure.

The MILP model also relied on the deterministic estimation of operation times to ascertain the objective function value.

Transportation time was excluded; this was revealed to have an influence on the idle time values in Section 9.7; the effects were tangible, but would not have affected the final schedule due to the uniform shift in results. However, a closer approximation to reality could be achieved by including this parameter in the model.

The dissimilarity measure was computed with the assumption that the time taken for one pin removal would be equal to that of one pin insertion (bidirectional). This was due to the simplicity of the task of both removing and inserting the dowel pin modules on the fixture design implemented. This assumption disregarded actions such as the clamping, unclamping, constructing and deconstructing for more complex fixture modules. A non-bidirectional fixture reconfiguration would yield an asymmetric distance matrix. This would increase the complexity of grouping and sequencing of parts on fixtures (Stage I and Stage II, respectively), since the direction of reconfiguration would have to be considered.

The research explored the conceptual implementation of a fixture manufacturing cell, but the focus of the scheduling method was placed on fixture reconfigurations only. The fixture inventory was predetermined and remained constant throughout the schedule. Prospective investigations of the concept should include the influence of fabricating new fixtures on the scheduling method, such that an optimal fixture inventory can be maintained (as discussed in Section 10.6). The results revealed that a higher fixture inventory is conducive to superior schedules, but the influence of holding costs would have to be incorporated with such a model.

The fixture design was not a primary objective of the research. However, the design implemented was rudimentary in comparison to most practical fixtures. The fixture resolution was limited by the base plate array; the design was suitable for two-dimensional parts only. It is recommended that a more comprehensive fixture design for three-dimensional parts be implemented; a dissimilarity measure that is capable of computing the comparisons between different module configurations must be developed thereof. Additive manufacturing was investigated as a leading facilitator of mass customisation. It is suggested that a fixture design with 3D-printed custom modules would represent a sensible progression for the research field.

The production system was limited to one cell per operation type, and unit workflow. The MILP model could be modified to handle more complex problems hereafter, as was the case for the job shop scheduling problem history [75]. Parallel cells or multi-operator cells would enhance the throughput and scheduling options available to the fixture-part mappings; similar improvements could be produced through a batch workflow policy.

The MILP model was found to be computationally expensive to solve. The solution time increased exponentially for increasing problem sizes; this limited the problem test samples to 12 parts. The limitation was expected due to: the exact method approach; and the use of the branch and bound solution technique, which is known to increase exponentially with problem size [113]. The limitation detracted from investigating large-sized problems expected for mass customisation. Furthermore, the Stage III Heuristic could not be adequately compared to a benchmark value for the larger-sized problems that it solved. The S3H proved that feasible solutions did exist for larger-sized problems. It is believed that a more specialised software for operations research would have yielded a larger optimal solution set; these software packages would include CPLEX®, GUROBI® and SCIP®, as investigated by Ku and

Beck [76] for MIP problems. MATLAB® was also unable to utilise parallel computing for the MILP solver, despite the quad-core processor of the machine employed for testing. Furthermore, early endeavours at formulating the MILP model attempted to use the Big M method [52] to deal with the successive flow of fixture-part mappings from Cell 1 to Cell 2, i.e. the purpose of Equation (6.7). However, the software used was unable to implement this approach through *intlinprog*. The resultant solution introduced an additional slack variable (Y_{ijklk}), and required a non-linear constraint to be implemented; this had to be linearised through additional constraints (Equation (6.7a) to (6.7c)). Thus, the complexity of the MILP model was increased due to the technical limitations of the solver. It is recommended that the Big M method be evaluated as an alternative to Equation (6.7), where valid pairs of fixture-part mappings could be assessed as a difference (instead of product) by introducing a very large number to ensure that non-active fixture-part mappings are rejected from these pairs.

The Stage III Heuristic relied on random selection of the first pivot element. This initial element (or fixture-part mapping, in context of the application) significantly influenced the final results, due to the greedy algorithm approach being applied from that element onwards. The solution quality varies depending on this initial pivot, but the comparison of every available option for large-sized problems could be exhaustive. This provides scope for improvement, by conceivably introducing local search methods (as discussed in Section 2.7.2) that could efficiently obtain the best initial pivot, from which a superior solution could be achieved.

The Lab FxMC was assembled and automated as a proof-of-concept. A comprehensive study on a similar cell, or a fully-automated cell of its type, could be conducted as a practical case study to accurately evaluate the industrial application of a fixture manufacturing cell.

Lastly, the optimality of the scheduling method was affected by separating the method into three stages, resulting in an accumulation of inaccuracies from each step. A brief investigation into the Vehicle Routing Problem (VRP) revealed that the formulation could be used to solve both Stage I and Stage II in one (optimal) step. The VRP involves dispatching vehicles to destinations, such that the total distance traversed by the vehicles is minimised [116]. The vehicles are analogous to fixtures, and the destinations are analogous to parts. Further investigation into the non-Euclidean instances would be required, as formulations of this type are less common than the direct application of the VRP for its conventional purpose. However, the VRP approach does appear to be a promising progression for the research, the solution of which may nullify several shortcomings described in this section.

10.8 The FxMC Implications

The on-demand fixture manufacturing cell was proposed as a solution for fixture management in a mass customisation production system. It was established that mass customisation is a research area of increasing importance, and that dedicated fixtures in these systems are not applicable. This necessitates the use of reconfigurable fixtures, such as the modular fixture employed in the research. It was noted that the traditional management of fixtures (of the dedicated type) involves outsourcing the fabrication of these fixtures to an off-site facility, which is not conducive to mass customisation. Mass customisation requires responsiveness and adaptability, which are hindered by the reliance on an external supplier. The FxMC acts as a dedicated fixture facility on the shop floor, where the fixturing needs of the manufacturing system can be attended to with quick responsiveness. Furthermore, the

adaptability of the fixtures employed requires the FxMC activities to be co-ordinated with the production of the customised parts. The focus of this research was on the reconfiguration of the fixture modules for this activity. The research gap revealed that current scheduling and fixture management approaches have not catered to this need for mass customisation. The FxMC concept, in itself, provides a viable solution to the problem.

The research developed a CM system based on the concept of a centralised cell specialised for fixtures, i.e. the FxMC. A scheduling method was developed, which managed the activities in this cell such that the production system could be efficiently served with fixtures for customised parts. The benefits of the scheduling method developed are inherent in the time savings obtained from minimised idle time of Stage III. This implicitly reduces makespan and directly improves machine utilisation. The functions of Stage I and II also ensure minimised operation time for the reconfiguration task, which increases the likelihood of reduced idle times. The two-cell JIT workflow was based on maximum fixture utilisation and minimum fixture resource use, due to the lack of buffering in the system. A production system can benefit from the increased utilisation of fixtures, due to the ability of the fixtures to recycle through the FxMC and be returned in time for the incoming product. The concept, then, can help reduce the holding costs associated with fixture inventory. Thus, the benefits of low WIP through a pull production system is observed. The disadvantages of low WIP, discussed in Section 5.2, can be circumvented by the ability of the cell to fabricate new fixtures on-demand. This is a feature not catered to by the scheduling method, but a capability of the cell layout formulated for the research. The internal capability of a manufacturing system to increase its fixture inventory is conducive to the RMS paradigm, where scalability is a fundamental characteristic. This provides a manufacturing facility with the flexibility to handle a variety of product types, or change its target market according to economic needs and market trends. This aligns with the responsiveness and adaptability required for mass customisation.

The implementation of the FxMC would be met with similar disadvantages to that of CM systems (see Section 2.5). Achieving the advantages of the cell would necessitate restructuring of the shop floor layout. This would lead to downtime that would impede productivity for that period. The cost implication would have to be evaluated before such an undertaking can be performed. The concept of a Virtual Manufacturing Cell (VMC) could provide a solution to this problem. A VMC does not necessitate the physical restructuring of shop floor equipment; the VMC routes the part family associated with it across the resources that it comprises of on the current shop floor [38]. This would reduce the optimality and efficiency in comparison to a dedicated cell, but may be a necessary compromise for manufacturing facilities with constraints that would prevent the physical conversion to a CM system (such as a lack of floor space).

Other disadvantages of the FxMC implementation could include: investment in new equipment and resources dedicated to the cell; reduced fixture performance (compared to dedicated fixtures) due to modular components; complex PPC system for managing both parts and fixtures in-tandem; and staff training necessary for operating a new system.

The research outcomes revealed a manufacturing system with high fixture and machine utilisation, low WIP, reduced buffering costs, and quick responsiveness to changing product types. These were shown to be conducive to mass customisation and reconfigurable fixtures. The on-demand fixture manufacturing cell, then, can facilitate the manufacture of customised products.

10.9 Summary

This chapter discussed the research outcomes and findings. Each aspect of the research was addressed and evaluated in comparison to the research objectives. Shortcomings were noted and recommendations for future applications and developments were made. The implications of the fixture manufacturing cell was then discussed, from which it was decided that the concept does sufficiently address the problem it set out to investigate and solve.

11 Conclusion

11.1 Introduction

This chapter concludes the findings of the research outcomes in line with the aims and objectives of the research problem. The research contributions are discussed, before the recommendations for future work are summarised.

11.2 Research Aims and Objectives Evaluation

The research outcomes are evaluated in comparison to the aims and objectives set out in Section 1.4.

The first three objectives were fulfilled through the conceptualisation of the FxMC and its layout, and the assembly and automation of the proof-of-concept. The RMS, GT and CM paradigms were investigated and implemented in the research. A production system was formulated from the characteristics of the paradigms investigated; this was a two-cell JIT cellular manufacturing system with unit workflow, which was employed as a representation of a mass customisation production system. The Lab FxMC was assembled and automated through the use of a relay circuit, PC and PLC control. A pneumatic system with sensor was implemented with the PLC. MATLAB®, Arduino®, ladder logic (for PLC) and G-code (for CNC router) were either implemented or modified for the automation of the fixture manufacturing cell.

The fourth objective was fulfilled through the emphasis placed on the development of an optimisation model for the scheduling of the fixture manufacturing cell. A three-stage method was created, which: assigned parts to fixtures; sequenced parts on those fixtures; and scheduled fixture and part operations in the two-cell system, such that total idle time was minimised. The result was a schedule that reduced the makespan of the job list (through minimised total idle time) and maximised cell utilisation. The MILP model developed had limitations, which were addressed through an alternative heuristic; this heuristic generated feasible, near-optimal solutions aligned with the original objective function.

The fifth objective was fulfilled through the development of an agent-based simulation. A simulation of the FxMC was developed in AnyLogic®, with supplementary Java® commands for specific modelling of the process flowchart to correspond to the cell behaviour.

The sixth objective was fulfilled through the extensive testing of the research outcomes in Chapter 9. The findings were summarised as follows: the MDS procedure was robust for larger problems; the dissimilarity measure and clustering algorithm were validated; the intracluster sequence was the optimal such sequence for the respective group; the MILP model reliably solved problems with up to 12 parts to optimality; The S3H produced feasible, near-optimal solutions, which confirmed the computational limitations of the MILP solver used; the simulation revealed the effect of transportation time on the schedule, and provided real-time performance metrics for the job list execution; high levels of utilisation were observed due to the minimisation of idle times.

These objectives, and the completion of them, ensured that the aim of developing an on-demand fixture manufacturing cell and optimal scheduling method for mass customisation production systems was achieved.

11.3 Research Contribution

The research gap was indicated, where the scheduling of fixtures in a manufacturing system was not addressed at a comprehensive level thus far. The research outcomes provided contributions toward the field of scheduling and optimisation within the mass customisation and reconfigurable fixture research areas.

The concept of the fixture manufacturing cell as a solution for the management of reconfigurable fixtures in a mass customisation production system was proposed.

A fixture manufacturing cell layout was proposed; the cell was assembled and automated as a proof-of-concept.

The implementation of the fixture manufacturing cell within the production planning and control system was outlined.

A scheduling method for the circulation of reconfigurable fixtures in a manufacturing system was developed; this included: a dissimilarity measure, clustering technique, and sequencing technique for reconfigurable modular fixtures; and a MILP model for the optimal scheduling of a two-cell JIT system.

A heuristic was created as a near-optimal substitute for the aforementioned MILP model.

An agent-based simulation of the fixture manufacturing cell was developed for testing of the production characteristics and performance.

The research provided a platform upon which scheduling of reconfigurable fixtures for customised parts can be expanded and benchmarked. The scheduling method minimised the total idle time, which implicitly minimised the makespan. The results showed that minimised idle times lead to high utilisation (as expected).

The research outcomes stand to benefit manufacturing enterprises that utilise reconfigurable fixtures, preferably through a cellular manufacturing system. The MILP model developed relies only on the operation times used; thus, the model can be used to schedule any two-cell JIT manufacturing system for minimisation of total idle time. The benefits of the optimal schedule include: lower WIP due to the lean manufacturing operation of the JIT system; improved utilisation and flexibility of both machines and fixtures; customer satisfaction through reliable and punctual product delivery. These benefits are in addition to the competitive advantages of mass customisation and the efficiency improvements of cellular manufacturing systems.

11.4 Future Work

Research avenues for future work were suggested amongst the recommendations in Section 10.7. A fixture design for three-dimensional parts, with 3D-printed custom modules, could be implemented; the development of an appropriate dissimilarity measure would be required thereof. A stochastic modelling approach to the problem, or extending the current approach through a metaheuristic, could be developed. The influence of fabricating new fixtures, and incorporating this into the model, could be investigated. The application of the scheduling method could be expanded to parallel cells and batch workflow. The model could be re-evaluated with specialised operations research software; thus, the Y_{ijkl} slack variable could be eliminated by using the Big M method, which should simplify the MILP model. A fully-automated cell could be implemented for an industrial case study to gain further insight into the influence of the fixture manufacturing cell on a mass customisation system. Lastly, the VRP could be investigated as a potential approach to simultaneously solving both Stage I and Stage II with superior efficiency and optimality.

11.5 Summary

The aims and objectives stated in Section 1.4 were fulfilled by the research outcomes discussed throughout the dissertation. The research contributions were noted and the advantages of their applications were summarised. A summary of future work recommendations derived from Section 10.7 was presented.

It can be concluded that the on-demand fixture manufacturing cell concept provides a promising technique for the management of reconfigurable fixtures in a mass customisation production system. Through the implementation of RMS and CM concepts, a mass customisation production system can be implemented with improvements in efficiency, flexibility, responsiveness, and delivery in comparison to job shops and dedicated manufacturing lines. The fixture manufacturing cell can facilitate those advantages through the distribution and management of reconfigurable fixtures via a specialised and centralised cell. Thus, the aim and objectives were met, from which the research question (Section 1.3) can be answered as yes: an on-demand fixture manufacturing cell, with suitable production planning and control, can facilitate the manufacture of customised products.

References

- [1] X. Yao and Y. Lin, "Emerging manufacturing paradigm shifts for the incoming industrial revolution," *Int. J. Adv. Manuf. Technol.*, vol. 85, no. 5, pp. 1665–1676, 2016.
- [2] F. S. Fogliatto, G. J. C. da Silveira, and D. Borenstein, "The mass customization decade : An updated review of the literature," *Int. J. Prod. Econ.*, vol. 138, no. 1, pp. 14–25, 2012.
- [3] A. Y. C. Nee, Z. J. Tao, and A. V. Senthil Kumar, "Introduction to Fixture Design," in *An Advanced Treatise on Fixture Design and Planning*, 1st ed., Singapore: World Scientific Publishing Co., 2004, pp. 1–20.
- [4] Z. M. Bi and W. J. Zhang, "Flexible fixture design and automation: Review, issues and future directions," *Int. J. Prod. Res.*, vol. 39, no. 13, pp. 2867–2894, 2001.
- [5] T. O. Kowang, N. Mohd Hamel, C. S. Long, and A. Mohd Rasli, "Operation Management: Project Management in Jig and Fixture Industries," *Adv. Mater. Res.*, vol. 931–932, pp. 1621–1625, 2014.
- [6] B. Esmailian, S. Behdad, and B. Wang, "The evolution and future of manufacturing: A review," *J. Manuf. Syst.*, vol. 39, pp. 79–100, 2016.
- [7] K. Thörnblad, A.-B. Strömberg, M. Patriksson, and T. Almgren, *Scheduling optimization of a real flexible job shop including side constraints regarding maintenance, fixtures, and night shifts*. Department of Mathematical Sciences, Division of Mathematics, Chalmers University of Technology and University of Gothenburg, 2013.
- [8] T. C. Wong, F. T. S. Chan, and L. Y. Chan, "A resource-constrained assembly job shop scheduling problem with Lot Streaming technique," *Comput. Ind. Eng.*, vol. 57, no. 3, pp. 983–995, 2009.
- [9] J.-M. Yu, H.-H. Doh, J.-S. Kim, D.-H. Lee, and S.-H. Nam, "Scheduling for a Reconfigurable Manufacturing System with Multiple Process Plans and Limited Pallets/Fixtures," *Int. J. Mech. Aerospace, Ind. Mechatron. Manuf. Eng.*, vol. 6, no. 2, pp. 232–237, 2012.
- [10] H.-H. Doh, J.-M. Yu, and D.-H. Lee, "Priority Scheduling for a Flexible Job Shop with a Reconfigurable Manufacturing Cell," *Ind. Eng. Manag. Syst.*, vol. 15, no. 1, pp. 11–18, 2016.
- [11] O. J. Bakker, T. Papastathis, S. Ratchev, and A. A. . Popov, "Recent research on flexible fixtures for manufacturing processes," *Recent Patents Mech. Eng.*, vol. 6, no. 2, pp. 107–121, 2013.
- [12] Z. M. Bi, S. Y. T. Lang, M. Verner, and P. Orban, "Development of reconfigurable machines," *Int. J. Adv. Manuf. Technol.*, vol. 39, no. 11–12, pp. 1227–1251, 2008.
- [13] F. T. Piller, K. Moeslein, and C. M. Stotko, "Does mass customization pay? An economic approach to evaluate customer integration," *Prod. Plan. Control*, vol. 15, no. 4, pp. 435–444, 2004.
- [14] S. Smith, R. Jiao, and C.-H. Chu, "Editorial: Advances in mass customization," *J. Intell. Manuf.*, vol. 24, no. 5, pp. 873–876, 2013.
- [15] S. Chen, Y. Wang, and M. M. Tseng, "Mass customisation as a collaborative engineering effort," *Int. J. Collab. Eng.*, vol. 1, pp. 1–28, 2009.

- [16] B. Squire, S. Brown, J. Readman, and J. Bessant, "The impact of mass customisation on manufacturing trade-offs," *Prod. Oper. Manag.*, vol. 15, no. 1, pp. 10–21, 2006.
- [17] J. M. Tien, "Data mining requirements for customized goods and services," *Int. J. Inf. Technol. Decis. Mak.*, vol. 5, no. 4, pp. 693–698, 2006.
- [18] J. H. Mikkola and T. Skjøtt-Larsen, "Supply-chain integration: implications for mass customization, modularization and postponement strategies," *Prod. Plan. Control*, vol. 15, no. 4, pp. 352–361, 2004.
- [19] M. H. Meyer, "Revitalise your product lines through continuous platform renewal," *Res. Technol. Manag.*, vol. 40, no. 2, pp. 17–28, 1997.
- [20] J. K. Gershenson, G. J. Prasad, and Y. Zhang, "Product modularity: Definitions and benefits," *J. Eng. Des.*, vol. 14, no. 3, pp. 295–313, 2003.
- [21] H. Skipworth and A. Harrison, "Implications of form postponement to manufacturing: a case study," *Int. J. Prod. Res.*, vol. 42, no. 10, pp. 2063–2081, 2004.
- [22] H. Wang, Y. Rong, H. Li, and P. Shaun, "Computer aided fixture design: Recent research and trends," *CAD Comput. Aided Des.*, vol. 42, no. 12, pp. 1085–1094, 2010.
- [23] H. Hashemi, A. M. Shaharoun, and S. Izman, "Fixture designers guidance: A review of recent advanced approaches," *Jordan J. Mech. Ind. Eng.*, vol. 8, no. 6, pp. 377–384, 2014.
- [24] T. Kow, A. Kumar, and J. Fuh, "An integrated approach to collision-free computer-aided modular fixture design," *Int. J. Adv. Manuf. Technol.*, vol. 16, no. 4, pp. 233–242, 2000.
- [25] J. F. Hurtado and S. N. Melkote, "A model for synthesis of the fixturing configuration in pin-array type flexible machining fixtures," *Int. J. Mach. Tools Manuf.*, vol. 42, no. 7, pp. 837–849, 2002.
- [26] A. Gameros, S. Lowth, D. Axinte, A. Nagy-Sochacki, O. Craig, and H. R. Siller, "State-of-the-art in fixture systems for the manufacture and assembly of rigid components: A review," *Int. J. Mach. Tools Manuf.*, vol. 123, pp. 1–21, 2017.
- [27] R. Müller, M. Esser, and M. Vette, "Reconfigurable handling systems as an enabler for large components in mass customized production," *J. Intell. Manuf.*, vol. 24, no. 5, pp. 977–990, 2013.
- [28] A. S. Wallack and J. F. Canny, "Planning for modular and hybrid fixtures," in *IEEE International Conference on Robotics and Automation*, 1994, pp. 520–527.
- [29] Y. Kang, Y. Rong, and J. C. Yang, "Computer-Aided Fixture Design Verification. Part 1. The Framework and Modelling," *Int. J. Adv. Manuf. Technol.*, vol. 21, no. 10–11, pp. 827–835, 2003.
- [30] Y. Kang, Y. Rong, J. Yang, and W. Ma, "Computer-aided fixture design verification," *Assem. Autom.*, vol. 22, no. 4, pp. 350–359, 2002.
- [31] Y. Koren *et al.*, "Reconfigurable Manufacturing Systems," *CIRP Ann. - Manuf. Technol.*, vol. 48, no. 2, pp. 527–540, 1999.
- [32] D. E. Toni and S. Tonchia, "Manufacturing flexibility: a literature review," *Int. J. Prod. Res.*, vol. 36, no. 6, pp. 587–617, 1998.

- [33] Y. Koren and M. Shpitalni, "Design of reconfigurable manufacturing systems," *J. Manuf. Syst.*, vol. 29, no. 4, pp. 130–141, 2010.
- [34] Y. Koren and A. G. Ulsoy, "Vision, principles and impact of reconfigurable manufacturing systems," *Powertrain International*, vol. 5, no. 3, pp. 14–21, 2002.
- [35] T.-C. Chang, R. A. Wysk, and H.-P. Wang, "Group Technology," in *Computer-Aided Manufacturing*, 2nd ed., W. J. Fabrycky and J. H. Mize, Eds. New Jersey: Prentice-Hall, 1998, pp. 471–514.
- [36] M. P. Groover, "Group Technology and Cellular Manufacturing," in *Automation, Production Systems, and Computer-Integrated Manufacturing*, Second Edi., West Conshohocken, PA: Prentice-Hall, 2001, pp. 420–459.
- [37] R. Timings and S. Wilkinson, "Group Technology and Flexible Manufacturing Systems," in *E-manufacturing - Applications of Advanced Technology to Manufacturing Processes*, 1st ed., Essex: Pearson Education, 2003, pp. 176–185.
- [38] T. W. Liao, "Group Technology and Cellular Manufacturing," in *Exploring Advanced Manufacturing Technologies*, 1st ed., S. Krar and A. Gill, Eds. New York: Industrial Press, 2003, pp. 10-3-1-10-3–11.
- [39] A. Gill, "Lean Manufacturing," in *Exploring Advanced Manufacturing Technologies*, 1st ed., S. Krar and A. Gill, Eds. New York: Industrial Press, 2003, pp. 10-2-1-10-2–8.
- [40] M. S. Daskin and K. L. Maass, "The p-Median Problem," in *Location Science*, 1st ed., G. Laporte, S. Nickel, and F. S. da Gama, Eds. Springer, 2015, pp. 21–45.
- [41] M. Bazargan-Lari, H. Kaebernick, and A. Harraf, "Cell formation and layout designs in a cellular manufacturing environment a case study," *Int. J. Prod. Res.*, vol. 38, no. 7, pp. 1689–1709, 2000.
- [42] M. P. Groover, "Production Planning and Control Systems," in *Automation, Production Systems, and Computer-Integrated Manufacturing*, Second Edi., New Jersey: Prentice-Hall, 2001, pp. 796–831.
- [43] W. J. Hopp and M. L. Spearman, "To Pull or Not to Pull: What Is the Question?," *Manuf. Serv. Oper. Manag.*, vol. 6, no. 2, pp. 133–148, 2004.
- [44] G. Gallego, "Material Requirements Planning (MRP)," *Columbia University IEOR 4000: Project Management*, 2003. [Online]. Available: http://www.columbia.edu/~gmg2/4000/pdf/lect_06.pdf. [Accessed: 02-Dec-2017].
- [45] C. A. Ptak, "MRP, MRP II, OPT, JIT, and CIM - Succession, Evolution, or Necessary Combination," *Prod. Invent. Manag. J.*, vol. 32, no. 2, pp. 7–11, 1991.
- [46] C. S. Kumar and R. Panneerselvam, "Literature review of JIT-Kanban system," *Int. J. Adv. Manuf. Technol.*, vol. 32, no. 3–4, pp. 393–408, 2007.
- [47] K. Brown and T. Mitchell, "A comparison of just in time and batch manufacturing the role of performance obstacles," *Acad. Manag. J.*, vol. 34, no. 4, pp. 906–917, 1991.
- [48] F. Giordano and M. M. Schiraldi, "On Just-In-Time Production Leveling," in *Operations Management*, M. M. Schiraldi, Ed. InTech, 2013, pp. 141–162.

- [49] B. J. Blair, "A review of the kanban production control research literature," *Prod. Oper. Manag.*, vol. 1, no. 4, pp. 393–411, 1992.
- [50] Kekre and Karmarker, "Batching policy in kanban system," *J. Manuf. Syst.*, vol. 8, pp. 317–328, 1989.
- [51] M. P. Groover, "Lean Production and Agile Manufacturing," in *Automation, Production Systems, and Computer-Integrated Manufacturing*, Second., West Conshohocken, PA: Prentice-Hall, 2001, pp. 832–845.
- [52] J. Bisschop, *Optimization Modeling*. Diakenhuisweg: Aimms, 2016.
- [53] X. S. Yang, "Mathematical Optimization," in *Introduction to Mathematical Optimization - From Linear Programming to Metaheuristics*, Cambridge: Cambridge International Science Publishing, 2008, pp. 3–10.
- [54] H. G. Beyer and B. Sendhoff, "Robust optimization - A comprehensive survey," *Comput. Methods Appl. Mech. Eng.*, vol. 196, no. 33–34, pp. 3190–3218, 2007.
- [55] D. Ouelhadj and S. Petrovic, "A survey of dynamic scheduling in manufacturing systems," *J. Sched.*, vol. 12, no. 4, pp. 417–431, 2009.
- [56] V. Béla, "The Branch and Bound Method," in *Algorithms on Informatics*, 3rd ed., vol. 3, A. Iványi, Ed. Budapest: AnTonCom, 2011, pp. 1208–1261.
- [57] M. L. Pinedo, "Mathematical Programming: Formulations and Applications," in *Scheduling - Theory, Algorithms, and Systems*, 3rd ed., New York: Springer, 2008, pp. 559–571.
- [58] E. L. Lawler and D. E. . Wood, "Branch-And-Bound Methods : A Survey," *INFORMS Stable*, vol. 14, no. 4, pp. 699–719, 1966.
- [59] M. L. Pinedo, "Deterministic and Stochastic Dynamic Programming," in *Scheduling - Theory, Algorithms, and Systems*, 3rd ed., New York: Springer, 2008, pp. 573–579.
- [60] R. Kolisch and S. Hartmann, "Heuristic Algorithms for the Resource-Constrained Project Scheduling Problem: Classification and Computational Analysis," in *Project Scheduling - Recent Models, Algorithms and Applications*, vol. 14, New York: Springer, 1999, pp. 147–178.
- [61] P. Brucker, A. Drexl, R. Möhring, K. Neumann, and E. Pesch, "Resource-constrained project scheduling: Notation, classification, models, and methods," *Eur. J. Oper. Res.*, vol. 112, no. 1, pp. 3–41, 1999.
- [62] A. Manikas and Y.-L. Chang, "Multi-criteria sequence-dependent job shop scheduling using genetic algorithms," *Comput. Ind. Eng.*, vol. 56, no. 1, pp. 179–185, 2009.
- [63] M. Gen and L. Lin, "Multiobjective evolutionary algorithm for manufacturing scheduling problems: State-of-the-art survey," *J. Intell. Manuf.*, vol. 25, no. 5, pp. 849–866, 2014.
- [64] D. Lei, "Multi-objective production scheduling: A survey," *Int. J. Adv. Manuf. Technol.*, vol. 43, no. 9–10, pp. 925–938, 2009.
- [65] Y. Sun, C. Zhang, L. Gao, and X. Wang, "Multi-objective optimization algorithms for flow shop scheduling problem: A review and prospects," *Int. J. Adv. Manuf. Technol.*, vol. 55, no. 5–8, pp. 723–739, 2011.

- [66] M. L. Pinedo, "General Purpose Procedures for Deterministic Scheduling," in *Scheduling - Theory, Algorithms, and Systems*, 3rd ed., New York: Springer, 2008, pp. 371–394.
- [67] A. Allahverdi, C. T. Ng, T. C. E. Cheng, and M. Y. Kovalyov, "A survey of scheduling problems with setup times or costs," *Eur. J. Oper. Res.*, vol. 187, no. 3, pp. 985–1032, 2008.
- [68] S. Nguyen, Y. Mei, H. Ma, A. Chen, and M. Zhang, "Evolutionary Scheduling and Combinatorial Optimisation: Applications, Challenges, and Future Directions," in *2016 IEEE Congress on Evolutionary Computation*, 2016, pp. 3053–3060.
- [69] R. F. Tavares Neto and M. Godinho Filho, "Literature review regarding Ant Colony Optimization applied to scheduling problems: Guidelines for implementation and directions for future research," *Eng. Appl. Artif. Intell.*, vol. 26, no. 1, pp. 150–161, 2013.
- [70] G. Koulinas, L. Kotsikas, and K. Anagnostopoulos, "A particle swarm optimization based hyper-heuristic algorithm for the classic resource constrained project scheduling problem," *Inf. Sci. (Ny)*, vol. 277, pp. 680–693, Sep. 2014.
- [71] N. Bhatt and N. R. Chauhan, "Genetic algorithm applications on Job Shop Scheduling Problem: A review," in *2015 International Conference on Soft Computing Techniques and Implementations (ICSCTI)*, 2015, pp. 7–14.
- [72] G. H. Brooks and C. R. White, "An algorithm for finding optimal or near-optimal solutions to the production scheduling problem," *J. Ind. Eng.*, vol. 16, no. 1, pp. 34–40, 1965.
- [73] Y. Demir and S. Kürşat İşleyen, "Evaluation of mathematical models for flexible job-shop scheduling problems," *Appl. Math. Model.*, vol. 37, no. 3, pp. 977–988, 2013.
- [74] M. L. Pinedo, "Introduction," in *Scheduling - Theory, Algorithms, and Systems*, 3rd ed., New York: Springer, 2008, pp. 1–10.
- [75] J. Blazewicz, W. Domschke, and E. Pesch, "The job shop scheduling problem: Conventional and new solution techniques," *Eur. J. Oper. Res.*, vol. 93, no. 1, pp. 1–33, 1996.
- [76] W.-Y. Ku and J. C. Beck, "Mixed Integer Programming models for job shop scheduling: A computational analysis," *Comput. Oper. Res.*, vol. 73, pp. 165–173, 2016.
- [77] A. S. Manne, "On the Job-Shop Scheduling Problem," *Oper. Res.*, vol. 8, no. 2, pp. 219–223, 1960.
- [78] M. L. Pinedo, "Job Shops (Deterministic)," in *Scheduling - Theory, Algorithms, and Systems*, 3rd ed., New York: Springer, 2008, pp. 179–216.
- [79] J. Kuhpfahl, "Job Shop Scheduling - Formulation and Modeling," in *Job Shop Scheduling with Consideration of Due Dates - Potentials of Local Search Based Solution Techniques*, 1st ed., Halle: Springer Gabler, 2016, pp. 9–18.
- [80] E. G. Birgin, J. E. Ferreira, and D. P. Ronconi, "List scheduling and beam search methods for the flexible job shop scheduling problem with sequencing flexibility," *Eur. J. Oper. Res.*, vol. 247, no. 2, pp. 421–440, 2015.
- [81] R. Mencia, M. R. Sierra, C. Mencia, and R. Varela, "A genetic algorithm for job-shop scheduling with operators enhanced by weak Lamarckian evolution and search space narrowing," *Nat. Comput.*, vol. 13, no. 2, pp. 179–192, 2014.

- [82] A. Jalilvand-Nejad and P. Fattahi, "A mathematical model and genetic algorithm to cyclic flexible job shop scheduling problem," *J. Intell. Manuf.*, vol. 26, no. 6, pp. 1085–1098, 2015.
- [83] M. Sakhaii, R. Tavakkoli-Moghaddam, M. Bagheri, and B. Vatani, "A robust optimization approach for an integrated dynamic cellular manufacturing system and production planning with unreliable machines," *Appl. Math. Model.*, vol. 40, no. 1, pp. 169–191, Jan. 2016.
- [84] C. Liu, J. Wang, J. Y.-T. Leung, and K. Li, "Solving cell formation and task scheduling in cellular manufacturing system by discrete bacteria foraging algorithm," *Int. J. Prod. Res.*, vol. 54, no. 3, pp. 923–944, 2016.
- [85] R. Raminfar, N. Zulkifli, M. Vasili, and T. Sai Hong, "An integrated model for production planning and cell formation in cellular manufacturing systems," *J. Appl. Math.*, vol. 2013, pp. 1–10, 2013.
- [86] B. J. V. da Silva, R. Morabito, D. S. Yamashita, and H. H. Yanasse, "Production scheduling of assembly fixtures in the aeronautical industry," *Comput. Ind. Eng.*, vol. 67, pp. 195–203, 2014.
- [87] D. Eyers and K. Dotchev, "Technology review for mass customisation using rapid manufacturing," *Assem. Autom.*, vol. 30, no. 1, pp. 39–46, 2010.
- [88] ASTM International, *F2792-12a - Standard Terminology for Additive Manufacturing Technologies*. West Conshohocken, PA: ASTM, 2013.
- [89] S. H. Huang, P. Liu, A. Mokasdar, and L. Hou, "Additive manufacturing and its societal impact: A literature review," *Int. J. Adv. Manuf. Technol.*, vol. 67, no. 5–8, pp. 1191–1203, 2013.
- [90] P. Reeves, "How the socioeconomic benefits of rapid manufacturing can offset technological limitations," in *RAPID Conference and Exposition*, 2008, pp. 1–12.
- [91] C. Tuck, R. Hague, and N. Burns, "Rapid manufacturing: Impact on supply chain methodologies and practice," *Int. J. Supply Oper. Manag.*, vol. 3, no. 1, pp. 1–22, 2007.
- [92] E. Oberg, F. D. Jones, H. L. Horton, and H. H. Ryffel, "Machining Operations - Speed and Feed Tables," in *Machinery's Handbook*, 29th ed., C. J. McCauley, Ed. New York: Industrial Press, 2012, pp. 1021–1080.
- [93] M. P. Groover, "Cutting-Tool Technology," in *Fundamentals of Modern Manufacturing*, 4th ed., John Wiley & Sons, 2010.
- [94] H. A. El Maraghy, "Flexible and reconfigurable manufacturing systems paradigms," *Flex. Serv. Manuf. J.*, vol. 17, no. 4, pp. 261–276, 2006.
- [95] P. Croser and F. Ebel, *Pneumatics*, 10/2002. Denkendorf: Festo Didactic GmbH & Co., 2002.
- [96] W. Bolton, "Ladder and functional block programming," in *Programmable Logic Controllers*, 4th ed., Burlington: Elsevier Newnes, 2006, pp. 80–107.
- [97] A. J. Walker and G. Bright, "Stabilisation and control of configurable product manufacturing through Biased Decision Feedback decoupling," *J. Manuf. Syst.*, vol. 32, no. 1, pp. 271–280, 2013.
- [98] M. P. Groover, "Product Design and CAD/CAM in the Production System," in *Automation, Production Systems, and Computer-Integrated Manufacturing*, Second Edi., West Conshohocken, PA: Prentice-Hall, 2001, pp. 753–774.

- [99] M. P. Groover, "Process Planning and Concurrent Engineering," in *Automation, Production Systems, and Computer-Integrated Manufacturing*, Second Edi., West Conshohocken, PA: Prentice-Hall, 2001, pp. 775–796.
- [100] E. Naidoo, J. Padayachee, and G. Bright, "Optimal Scheduling of an on-Demand Fixture Manufacturing Cell for Mass Customisation Production Systems - Model Formulation, Presentation and Validation," in *Proceedings of the 14th International Conference on Informatics in Control, Automation and Robotics*, 2017, vol. 1, pp. 17–24.
- [101] S.-S. Choi, S.-H. Cha, and C. C. Tappert, "A Survey of Binary Similarity and Distance Measures," *J. Syst. Cybern. Informatics*, vol. 8, no. 1, pp. 43–48, 2010.
- [102] L. Rokach, "A survey of Clustering Algorithms," in *Data Mining and Knowledge Discovery Handbook*, 2nd ed., L. Rokach and O. Maimon, Eds. New York: Springer, 2010, pp. 269–298.
- [103] I. Dokmanic, R. Parhizkar, J. Ranieri, and M. Vetterli, "Euclidean distance matrices," *IEEE Signal Processing Magazine*, vol. 2015, no. November, pp. 12–30, 2015.
- [104] J. B. Kruskal, "Nonmetric Multidimensional Scaling: A Numerical Method," *Psychometrika*, vol. 29, no. 2, pp. 115–129, 1964.
- [105] K. Sturrock and J. Rocha, "A Multidimensional Scaling Stress Evaluation Table," *Field methods*, vol. 12, no. 1, pp. 49–60, 2000.
- [106] S. P. Lloyd, "Least Squares Quantization in PCM," *IEEE Trans. Inf. Theory*, vol. 28, no. 2, pp. 129–137, 1982.
- [107] D. Arthur and S. Vassilvitskii, "K-Means++: the Advantages of Careful Seeding," in *Proceedings of the 18th annual ACM-SIAM symposium on Discrete algorithms*, 2007, pp. 1027–1025.
- [108] Z. Bar-Joseph, D. K. Gifford, and T. S. Jaakkola, "Fast optimal leaf ordering for hierarchical clustering," *Bioinformatics*, vol. 17, no. 1, pp. S22–S29, 2001.
- [109] L. Liberti, "Compact linearization for binary quadratic problems," *4OR*, vol. 5, no. 3, pp. 231–245, 2007.
- [110] B. A. McCarl and T. H. Spreen, "Linear Programming Modelling: Nonlinearities and Approximation," in *Applied Mathematical Programming Using Algebraic Systems*, 2011, pp. 9–19–21.
- [111] J. Knowles, "Branch and Bound," *University of Manchester*, 2015. [Online]. Available: <http://studentnet.cs.manchester.ac.uk/pgt/2014/COMP60342/COMP60342-2015-lec4.BranchandBound.pdf>. [Accessed: 16-Mar-2017].
- [112] E. Danna, E. Rothberg, and C. Le Pape, "Exploring relaxation induced neighborhoods to improve MIP solutions," *Math. Program.*, vol. 102, no. 1, pp. 71–90, 2005.
- [113] T. Ibaraki, "On the computational efficiency of branch-and-bound algorithms," *J. Oper. Res. Soc. Japan*, vol. 20, no. 1, pp. 16–35, 1977.
- [114] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math.*, vol. 20, pp. 53–65, 1987.

- [115] M. Matsumoto and T. Nishimura, “Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator,” *ACM Trans. Model. Comput. Simul.*, vol. 8, no. 1, pp. 3–30, 1998.
- [116] J.-F. Cordeau, G. Laporte, M. W. P. Savelsbergh, and D. Vigo, “Vehicle Routing,” in *Handbook in OR & MS*, vol. 14, no. 6, C. Barnhart and G. Laporte, Eds. Elsevier B.V., 2007, pp. 367–428.
- [117] Mean Well, “350W Single Output Switching Power Supply: NES-350 series,” 2015. [Online]. Available: <http://www.meanwell.com/productPdf.aspx?i=457>. [Accessed: 02-Dec-2017].
- [118] Huaguan Relays, “NT72(4459) & NT72-2,” 2015. [Online]. Available: <http://www.huaguan-relays.com/WebUpload/UpLoadFile/201582054330221.pdf>. [Accessed: 02-Dec-2017].
- [119] Festo, “Standard cylinder DSBC-32-500-PPSA-N3,” 2017. [Online]. Available: https://www.festo.com/cat/en-za_za/products_DSBC?CurrentIDCode1=DSBC-32-500-PPSA-N3&CurrentPartNo=1376478. [Accessed: 02-Dec-2017].
- [120] Festo, “One-way flow control valve GRLA-1/8-QS-6-D,” 2017. [Online]. Available: https://www.festo.com/cat/en-za_za/products_GRLA_QS?CurrentIDCode1=GRLA-1%2F8-QS-6-D&CurrentPartNo=193144. [Accessed: 02-Dec-2017].
- [121] Festo, “Solenoid valve VUVS-L20-M52-AD-G18-F7-1C1,” 2017. [Online]. Available: https://www.festo.com/cat/en-za_za/products_VUVS?CurrentIDCode1=VUVS-L20-M52-AD-G18-F7-1C1&CurrentPartNo=575263. [Accessed: 02-Dec-2017].
- [122] Festo, “Silencer AMTE-M-H-G18,” 2017. [Online]. Available: https://www.festo.com/cat/en-za_za/products_AMTE?CurrentIDCode1=AMTE-M-H-G18&CurrentPartNo=1206622. [Accessed: 02-Dec-2017].
- [123] Festo, “Diffuse light sensor SOEG-RT-M18-PA-K-2L,” 2017. [Online]. Available: https://www.festo.com/cat/en-za_za/products_SOEG_R?CurrentIDCode1=SOEG-RT-M18-PA-K-2L&CurrentPartNo=547912. [Accessed: 02-Dec-2017].

A. Appendix A: Testing Results

A.1. MDS Robustness

Table A.1 to Table A.10 present the Stress-1 values obtained for each test. The threshold values were obtained from the evaluation table of [105]. The tolerance percentage is the range of the tested value within the threshold value (positive for below threshold). The problem sizes tested are as stated in the table captions.

Table A.1: MDS test results (10 parts)

Test Run	Stress	Threshold	Tolerance %
1	0.1413	0.133	-6.2406
2	0.1413	0.133	-6.2406
3	0.1159	0.133	12.85714
4	0.1477	0.133	-11.0526
5	0.1162	0.133	12.63158
6	0.1398	0.133	-5.11278
7	0.1653	0.133	-24.2857
8	0.1561	0.133	-17.3684
9	0.1233	0.133	7.293233
10	0.1569	0.133	-17.9699

Table A.2: MDS test results (20 parts)

Test Run	Stress	Threshold	Tolerance %
1	0.2424	0.279	13.11828
2	0.2525	0.279	9.498208
3	0.2542	0.279	8.888889
4	0.2491	0.279	10.71685
5	0.2498	0.279	10.46595
6	0.257	0.279	7.885305
7	0.2441	0.279	12.50896
8	0.2371	0.279	15.01792
9	0.221	0.279	20.78853
10	0.2334	0.279	16.34409

Table A.3: MDS test results (30 parts)

Test Run	Stress	Threshold	Tolerance %
1	0.2731	0.328	16.7378
2	0.2868	0.328	12.56098
3	0.2729	0.328	16.79878
4	0.287	0.328	12.5
5	0.2843	0.328	13.32317
6	0.2954	0.328	9.939024
7	0.2738	0.328	16.52439
8	0.2728	0.328	16.82927
9	0.2672	0.328	18.53659
10	0.2795	0.328	14.78659

Table A.4: MDS test results (40 parts)

Test Run	Stress	Threshold	Tolerance %
1	0.3021	0.352	14.17614
2	0.306	0.352	13.06818
3	0.2919	0.352	17.07386
4	0.3008	0.352	14.54545
5	0.3049	0.352	13.38068
6	0.3036	0.352	13.75

7	0.2906	0.352	17.44318
8	0.2905	0.352	17.47159
9	0.3	0.352	14.77273
10	0.3008	0.352	14.54545

Table A.5: MDS test results (50 parts)

Test Run	Stress	Threshold	Tolerance %
1	0.3129	0.366	14.5082
2	0.316	0.366	13.6612
3	0.3107	0.366	15.10929
4	0.3044	0.366	16.8306
5	0.3164	0.366	13.55191
6	0.311	0.366	15.02732
7	0.3052	0.366	16.61202
8	0.307	0.366	16.12022
9	0.3115	0.366	14.89071
10	0.3108	0.366	15.08197

Table A.6: MDS test results (60 parts)

Test Run	Stress	Threshold	Tolerance %
1	0.3241	0.376	13.80319
2	0.3227	0.376	14.17553
3	0.318	0.376	15.42553
4	0.315	0.376	16.2234
5	0.3253	0.376	13.48404
6	0.3158	0.376	16.01064
7	0.3156	0.376	16.06383
8	0.3168	0.376	15.74468
9	0.3156	0.376	16.06383
10	0.3195	0.376	15.0266

Table A.7: MDS test results (70 parts)

Test Run	Stress	Threshold	Tolerance %
1	0.3313	0.384	13.72396
2	0.3316	0.384	13.64583
3	0.3238	0.384	15.67708
4	0.3242	0.384	15.57292
5	0.3321	0.384	13.51563
6	0.3267	0.384	14.92188
7	0.3249	0.384	15.39063
8	0.319	0.384	16.92708
9	0.3264	0.384	15
10	0.3288	0.384	14.375

Table A.8: MDS test results (80 parts)

Test Run	Stress	Threshold	Tolerance %
1	0.336	0.388	13.40206
2	0.3347	0.388	13.73711
3	0.3282	0.388	15.41237
4	0.3243	0.388	16.41753
5	0.3359	0.388	13.42784
6	0.3306	0.388	14.79381
7	0.3282	0.388	15.41237
8	0.3303	0.388	14.87113
9	0.3315	0.388	14.56186
10	0.3328	0.388	14.2268

Table A.9: MDS test results (90 parts)

Test Run	Stress	Threshold	Tolerance %
----------	--------	-----------	-------------

1	0.3391	0.392	13.4949
2	0.3405	0.392	13.13776
3	0.3349	0.392	14.56633
4	0.3291	0.392	16.04592
5	0.3384	0.392	13.67347
6	0.3369	0.392	14.05612
7	0.3304	0.392	15.71429
8	0.3363	0.392	14.20918
9	0.3365	0.392	14.15816
10	0.3393	0.392	13.44388

Table A.10: MDS test results (100 parts)

Test Run	Stress	Threshold	Tolerance
1	0.3422	0.396	13.58586
2	0.3437	0.396	13.20707
3	0.3382	0.396	14.59596
4	0.3323	0.396	16.08586
5	0.3407	0.396	13.96465
6	0.343	0.396	13.38384
7	0.3366	0.396	15
8	0.3413	0.396	13.81313
9	0.3416	0.396	13.73737
10	0.3406	0.396	13.9899

A.2. Silhouette Test (Constant Parts, Variable Fixtures)

Table A.11 to Table A.13 present the silhouette values obtained for the 100 parts, clustered in a variable number of fixtures. The order of these parts on the table are as per the clustering results, and thus undefined. The results are amalgamated and synthesised at the end of each table. The test samples are as stated in the table captions.

Table A.11: Silhouette tests; constant parts, variable fixtures (Sample 1)

	2f	4f	6f	8f	10f	12f	14f	16f	18f	20f
	-0.1101	0.1198	0.4777	0.4366	0.5230	0.7246	0.4762	0.4405	0.2833	0.2963
	0.4779	0.7797	0.4999	-0.0464	0.1157	0.1092	0.6273	0.5320	0.5320	0.5320
	0.5705	0.5973	0.0486	0.3648	0.7613	0.4501	0.2771	0.7227	0.7227	0.7227
	0.4301	0.7763	0.6730	0.3581	0.4770	0.6145	0.2333	0.2333	0.2333	0.2333
	0.5502	0.7284	0.8013	0.4463	0.3890	0.2549	0.6588	0.6588	0.6588	0.6588
	0.7313	0.6598	0.7911	0.8058	0.7759	0.7371	0.4342	0.4342	0.6520	0.6393
	0.5861	0.1357	0.2973	0.2052	0.6903	0.5096	0.6673	0.3642	0.3016	0.4748
	0.5904	0.3905	0.3822	0.3731	0.6514	0.6104	0.4365	0.4365	0.2499	0.4349
	0.4110	0.7034	0.6555	0.2690	0.3242	0.7157	0.3899	0.3899	0.3899	0.3899
	0.3955	0.5105	0.3634	0.8275	0.7677	0.7016	0.7407	0.7016	0.7362	0.6288
	0.6278	0.1351	0.4865	0.5606	0.8277	0.8458	0.5070	0.5070	0.6767	0.5188
	0.3650	0.7328	0.5610	0.5773	0.6547	0.6396	0.5234	0.5120	0.5120	0.5120
	0.0973	0.4058	0.0908	0.7505	0.7395	0.2349	0.3148	0.3148	0.3148	0.3148
	0.0279	0.2332	0.7450	0.6897	0.3968	0.3803	0.6145	0.7809	0.5096	0.4617
	0.4068	0.4816	0.1889	0.5900	0.5060	0.4203	0.3382	0.3382	0.3941	0.8709
	0.1409	0.5068	0.3977	0.2041	0.0527	0.7510	0.5515	0.4373	0.4710	0.5537
	0.4102	0.4621	0.2501	0.3749	0.6621	0.7187	0.7400	0.1407	0.1407	0.1407
	0.6762	0.6107	0.7363	0.7630	0.7630	0.7286	0.5500	0.5500	0.2855	0.6581
	0.5131	0.8041	0.1233	0.7114	0.2991	0.3700	0.2637	0.3505	0.2127	0.6455
	0.6662	0.6467	0.7648	0.8347	0.8149	0.6289	0.6664	0.6664	0.6664	0.6664
	0.1391	0.1439	0.8502	0.8383	0.2746	0.1640	0.5292	0.6907	0.6630	0.8085
	0.4934	0.7145	0.7095	0.6379	0.7483	0.4417	0.2752	0.2752	0.2752	0.2752
	0.6352	0.6684	0.3915	0.2787	0.5202	-0.0369	0.2772	0.6832	0.7083	0.8586
	0.7239	0.4608	0.6638	0.7342	0.8489	0.7937	0.7937	0.7832	0.7832	0.7929
	0.6024	0.7341	0.5317	0.5049	0.3501	0.3269	0.7886	0.7886	0.7886	0.7886
	0.4204	0.7853	0.1998	0.6424	0.3943	0.5458	0.0554	0.6253	0.5411	0.5898

	0.4060	0.4584	0.6809	0.4235	0.4587	0.8043	0.8579	0.5938	0.5334	0.2961
	0.5318	0.5083	0.5322	0.5866	0.7880	0.7752	0.7097	0.5747	0.7611	0.4215
	-0.0324	0.3074	0.3383	0.5230	0.4920	0.4659	0.5610	0.7450	0.7450	0.7450
	0.7466	0.5014	0.6037	0.4683	0.2907	0.1260	0.0778	0.0778	0.1456	0.3203
	0.5758	0.7940	0.7149	0.7838	0.3218	0.5724	0.4640	0.4161	0.4161	0.4161
	0.6607	0.5069	0.7522	0.4160	0.4290	0.3947	0.4540	0.6137	0.5348	0.7112
	0.3175	0.3626	0.7740	0.6038	0.4032	0.7024	0.8525	0.7060	0.7526	0.5966
	0.1360	0.5341	0.8164	0.0107	0.4471	0.8296	0.6730	0.8296	0.8694	0.6237
	0.7096	0.2548	0.2989	0.5475	0.5475	0.6052	0.6828	0.6828	0.6828	0.6828
	0.4736	0.6691	0.0496	0.7283	0.7744	0.7846	0.7278	0.7278	-0.1024	0.2427
	0.2706	0.2145	0.5963	0.7637	0.5639	0.3566	0.4532	0.3566	0.6388	0.4421
	0.2087	0.2574	0.6677	0.0973	0.0380	0.8265	0.8507	0.8507	0.8507	0.8507
	0.6550	0.8045	0.4044	0.5354	0.1530	0.5714	0.5714	0.3416	0.3416	0.3416
	0.5106	0.3175	0.4224	0.1675	0.8386	0.2183	0.1805	0.6391	0.5537	0.6445
	0.0597	0.0257	0.4866	0.5074	0.3874	0.7956	0.5649	0.5649	0.5649	0.5649
	0.2740	0.5846	0.3791	0.6164	0.6271	0.3930	0.7378	0.7378	0.7378	0.7378
	0.0640	0.5033	0.6050	0.0417	0.5374	0.2658	0.5875	0.3996	0.4950	0.4708
	0.4100	0.7164	0.3236	0.1721	0.1561	0.8267	0.5899	0.0060	0.5211	0.5492
	0.7082	0.7659	0.7999	0.7723	0.7063	0.6479	0.5201	0.5201	0.4597	0.3799
	0.1945	0.6447	0.7564	0.3911	0.7128	0.7213	0.7955	0.6672	0.1233	0.1301
	0.3007	0.4422	0.7157	0.1558	0.4404	0.6022	0.7188	0.7188	0.7188	0.7188
	0.7272	0.1823	0.3086	0.1412	0.1412	-0.0479	0.7771	0.7771	0.6879	0.6513
	0.5307	0.6413	0.3160	0.5240	0.6450	0.4610	0.5215	0.3771	0.3771	0.3771
	0.4603	0.0416	0.7696	0.7717	0.7819	0.7472	0.2284	-0.0846	0.2416	0.2115
	0.7297	0.1788	0.7871	0.8118	0.6636	0.7028	0.6754	0.6385	0.6385	0.3966
	0.3334	0.3838	0.5286	0.2933	0.3405	0.6716	0.7880	0.6712	0.6712	0.6712
	0.5393	0.7460	0.4777	0.3157	0.4087	0.4703	0.1639	0.1639	0.7357	0.8087
	0.7053	0.1841	-0.1302	0.1131	0.6091	0.6158	0.3996	0.3996	0.4146	0.5504
	0.6289	0.7533	0.5208	0.4014	0.7855	0.7890	0.8066	0.2924	-0.2839	-0.2686
	0.3712	0.6111	0.3982	0.4445	0.2584	0.0929	0.3823	0.2325	0.2325	0.2325
	0.7023	0.2318	0.4668	0.4184	0.1526	0.1862	0.0869	0.5908	0.5504	0.3604
	0.6907	0.5092	0.8153	0.4994	0.5156	0.4402	0.5109	0.4901	0.1361	0.4126
	0.2461	0.6444	0.7909	0.4003	0.5936	0.8165	0.7392	0.7812	0.8649	0.8183
	0.6985	0.4624	0.7064	0.3637	0.4438	0.3408	0.3778	0.2371	0.5481	0.7218
	0.4798	0.8368	0.4107	0.8297	0.1206	0.0834	0.0601	0.4183	0.0643	0.0451
	0.5875	0.7919	0.8217	0.5774	0.5051	-0.0046	0.5338	0.5338	0.5338	0.5338
	0.6955	0.4105	0.2072	0.3956	-0.0965	0.0287	0.6007	0.6007	0.5766	0.5374
	0.6007	0.5984	0.2285	0.4376	0.7214	0.7974	0.8168	0.7577	0.8286	0.5845
	0.2726	0.2697	0.6445	0.8380	0.7074	0.4637	0.6088	0.4637	0.6699	0.3675
	0.2384	0.5915	0.0730	0.4171	0.3813	0.4644	0.1268	0.4490	0.4490	0.4490
	-0.0675	0.3536	0.1393	0.4294	0.1740	0.6951	0.6550	0.8055	0.8055	0.8055
	0.6975	0.0696	0.8278	0.7887	0.4925	0.4557	0.3951	0.4380	0.4380	0.4602
	0.3324	0.4799	0.7977	0.3160	0.3319	0.2656	0.1044	0.0144	0.0144	0.0144
	0.0018	0.3580	0.8152	0.3367	0.0815	0.7062	0.4389	0.7062	0.6418	-0.1771
	0.5594	0.7884	0.6703	0.7013	0.7111	0.7620	0.7045	0.7045	0.7045	0.7045
	0.2718	0.6230	0.3712	0.7168	0.7399	0.4845	0.8253	0.8253	0.8253	0.8253
	0.6894	0.7281	0.8210	0.8253	0.8054	0.7900	0.8235	0.8235	0.2677	0.7311
	0.2067	0.1660	0.6836	0.0789	0.1993	0.0355	0.3728	0.5197	0.5197	0.5197
	-0.0682	0.1203	0.8045	0.7734	0.2035	0.0133	0.5039	0.5621	0.7029	0.6845
	0.6980	0.5322	0.4785	0.6521	0.4257	0.5225	0.6753	0.6753	0.6753	0.6721
	0.5274	0.6962	0.1927	0.5569	0.4634	0.3883	0.1920	0.1920	0.1827	0.7021
	0.6965	0.6936	0.2451	0.2836	0.6752	0.3057	0.3057	0.3700	0.3700	0.5774
	0.7108	0.1806	0.7805	0.8340	0.8009	0.7484	0.7241	0.8061	0.8061	0.2391
	0.6043	0.7894	0.7135	0.3543	0.2682	0.1912	0.5600	0.5600	0.7880	0.5269
	0.6383	0.7320	0.5311	0.3684	0.6917	0.5612	0.7002	0.4310	0.7420	0.8207
	0.6778	0.5644	0.0483	0.0837	0.6042	0.7203	0.7888	0.7888	0.7768	0.8240
	0.5828	0.7964	0.1790	0.6529	0.4346	0.3165	0.5241	0.6784	0.6997	0.6442
	0.3341	0.7247	0.7212	0.6155	0.5157	0.6739	0.6406	0.6356	0.7674	0.7068
	0.6510	0.7953	0.2930	0.3509	-0.0589	0.5375	0.5375	0.5184	0.5184	0.3590
	0.0971	0.1528	0.7673	0.6013	0.1097	0.3266	0.0540	0.3266	-0.0322	0.3120
	0.5122	0.7517	0.7999	0.7653	0.6277	0.6201	0.5097	0.5477	0.5477	0.5477
	0.7412	0.3555	0.5141	0.4412	0.4412	0.2895	0.6011	0.6011	0.3723	0.3253
	0.7079	0.4656	0.5682	0.7840	0.7840	0.7816	0.1872	0.1872	0.1872	0.1872

	0.6799	0.7811	0.6390	0.4394	0.1615	-0.0600	0.2632	0.2632	-0.1561	0.4373
	0.6224	0.8297	0.5935	0.6995	0.3716	0.7336	0.6854	0.5511	0.5511	0.5511
	0.6471	0.7328	0.7310	0.7373	0.6956	0.5386	0.2399	0.2399	0.2399	0.2399
	0.3817	0.3289	0.4283	0.2664	0.4386	0.3265	0.3547	0.0747	0.3361	0.3773
	0.2145	0.1809	0.5947	0.7135	0.5342	0.1331	0.3929	0.1331	0.1987	0.6242
	0.2543	0.6781	0.3414	0.5189	0.6626	0.3324	0.3084	0.1987	0.8362	0.8471
	0.7440	0.1398	0.1108	0.4909	0.4909	0.6420	0.6632	0.6632	0.6632	0.6632
	0.6434	0.8242	0.7801	0.4480	0.3538	0.2796	0.5706	0.5706	0.5639	0.3714
	0.5820	0.7369	0.3437	0.4226	0.6501	0.6906	0.6545	0.7101	0.5965	0.5659
	0.6468	0.6688	0.7853	0.7947	0.7706	0.4795	0.7946	0.7946	0.7946	0.7946
	0.6873	0.6135	0.3936	0.4808	0.7764	0.5115	0.5115	0.6225	0.6225	0.7103
Average	0.4690	0.5160	0.5205	0.5036	0.4955	0.4958	0.5183	0.5146	0.5055	0.5201
Median	0.5313	0.5493	0.5320	0.5021	0.5056	0.5300	0.5438	0.5505	0.5479	0.5498
SD	0.2286	0.2321	0.2399	0.2253	0.2330	0.2477	0.2170	0.2190	0.2518	0.2281
Min	-0.1101	0.0257	-0.1302	-0.0464	-0.0965	-0.0600	0.0540	-0.0846	-0.2839	-0.2686
Max	0.7466	0.8368	0.8502	0.8383	0.8489	0.8458	0.8579	0.8507	0.8694	0.8709

Table A.12: Silhouette tests; constant parts, variable fixtures (Sample 2)

	2f	4f	6f	8f	10f	12f	14f	16f	18f	20f
	-0.0338	0.1689	0.6766	0.7471	0.5366	0.4827	0.6331	0.4883	0.5877	0.3418
	0.7063	0.2630	0.8121	0.4489	-0.0602	0.0637	0.5106	0.3241	0.4329	0.6738
	0.7115	0.6931	0.3220	0.1399	0.2479	0.1266	-0.0745	0.3908	0.2361	-0.0153
	0.5637	0.7115	0.7314	0.3367	0.1985	0.2398	0.2398	0.1865	0.1865	1.0000
	0.0675	0.4121	0.8070	0.7357	0.4160	0.5712	0.6721	0.6984	0.5744	0.5744
	0.3509	0.5413	0.0507	0.4986	0.8223	0.2343	0.3738	0.3588	0.4032	0.4032
	0.6123	0.6968	0.3253	0.4666	0.6502	0.7019	0.7098	0.5428	0.7934	0.8539
	0.5381	0.7654	0.4641	0.7767	0.6597	0.6455	0.6455	0.6279	0.8056	0.7237
	0.5070	0.6815	0.3665	0.3020	0.2779	0.7535	0.7489	0.7357	0.7141	0.7164
	0.6545	0.4949	0.6361	0.3594	0.2585	0.3570	0.2638	0.5006	0.6651	0.3118
	0.5051	0.7884	0.5337	0.4568	0.4394	0.7510	0.5485	0.6651	0.6700	0.6700
	0.3910	0.5579	0.3551	0.4668	0.4300	0.8026	0.8056	0.3307	0.6163	0.3552
	0.7503	0.4074	0.3791	0.4727	0.2849	0.1194	0.5518	0.4769	0.2625	0.3648
	0.3835	0.1010	0.8136	0.8662	0.6896	0.6569	0.5768	0.6623	0.7147	0.6847
	0.7486	0.5307	0.5441	0.6883	0.6533	0.7445	0.5482	0.5936	0.2726	0.3423
	0.5592	0.7619	0.0355	0.6050	0.5510	0.7865	0.7865	0.8056	0.8342	0.8494
	0.4512	0.0739	0.1998	0.0075	0.6578	0.6821	0.1914	0.3207	0.4308	0.3188
	0.6852	0.5694	0.7329	0.4434	0.7685	0.7563	0.7631	0.7968	0.7962	0.4837
	0.6770	0.0815	0.5639	0.5103	0.7569	0.7237	0.7313	0.5740	0.6996	0.7710
	0.6209	0.7989	0.8124	0.7158	0.3264	0.3094	0.3699	0.4358	0.5449	0.5449
	0.7549	0.2914	0.2931	0.7063	0.4891	0.5133	0.2379	0.3410	0.2624	0.2282
	0.4137	0.6490	0.4229	0.6903	0.6869	0.6502	0.3102	0.4118	0.4785	0.4785
	0.6568	0.6772	0.7339	0.2749	0.6886	0.4443	0.5102	0.4853	0.4200	0.4200
	0.6664	0.5401	0.4507	0.3287	0.5664	0.2054	0.4399	0.2906	0.6009	0.4472
	0.7215	0.3764	0.3748	0.5513	0.5875	0.7027	0.7766	0.7621	0.7621	0.7621
	0.3416	0.7465	0.8087	0.7799	0.7282	0.0726	0.2911	0.2927	0.6531	0.6531
	-0.0312	0.1876	0.6950	0.5608	0.0362	0.4318	0.6645	0.7080	0.7080	0.7080
	0.3689	0.5232	0.2154	0.3346	0.5762	0.6907	0.6907	0.7195	0.6499	0.8663
	0.1596	0.5013	0.7539	0.3935	0.6182	0.6316	0.4617	0.3350	0.2737	0.3350
	0.5992	0.7759	0.0191	0.2584	0.4917	0.5271	0.7826	0.6158	0.6297	0.5887
	0.6445	0.8224	0.8224	0.7305	0.3231	0.3647	0.4857	0.4804	0.3927	0.3927
	0.4580	0.8152	0.6963	0.6448	0.5852	0.3303	0.3969	0.2718	0.4061	0.4061
	0.3365	0.3800	0.8271	0.7337	0.3757	0.2486	0.4353	0.4471	0.6566	0.6449
	0.5206	0.7520	0.0929	0.8075	0.6532	0.6822	0.5808	0.6392	0.6026	0.6456
	0.4268	0.5179	0.7511	0.4888	0.3093	0.1482	0.1364	0.1796	0.1180	0.1107
	0.7385	0.1020	0.2254	0.1430	0.4407	0.3967	0.8101	0.5760	0.8156	0.8503
	0.1864	0.1796	0.4521	0.0665	0.5331	0.4219	0.8317	0.8311	0.8423	0.8423
	0.6388	0.7354	0.2862	0.5924	0.6041	0.5704	0.7209	0.6901	0.6516	0.5268
	0.7095	0.7097	0.7517	0.2918	0.7260	0.7929	0.7929	0.6833	0.7125	0.6259
	0.6934	0.3746	0.7364	0.7749	0.6743	0.6267	0.6989	0.7467	0.6488	0.6672
	0.5477	0.7676	0.6266	0.5447	0.5935	0.5198	0.4221	0.1474	0.6281	0.6281
	0.1624	0.4440	0.7584	0.7950	0.7592	0.7483	0.8035	0.7848	0.8035	0.7848

	-0.0094	0.3564	0.0885	0.3667	0.2458	0.2713	0.4182	0.2185	0.2410	0.2410
	0.3397	0.7238	0.8326	0.8251	0.8065	0.4213	0.4550	0.4635	0.1636	0.1636
	0.4836	0.6891	0.6027	0.6859	0.5624	0.5513	0.5513	0.6056	0.4226	0.4338
	0.6554	0.6593	0.4389	0.2938	0.8091	0.7958	0.6600	0.3225	0.3225	0.3225
	0.6126	0.6685	0.3698	0.6785	0.3831	0.4606	0.3029	0.5042	0.6195	0.1887
	0.7354	0.3494	0.1110	0.2122	0.3056	0.1921	0.6899	0.2414	0.8508	0.8058
	0.3802	0.6736	0.3265	0.7069	0.6414	0.6387	0.6693	0.3013	0.0360	0.3946
	0.2585	0.3233	0.1127	0.0598	0.2088	0.5748	0.4834	0.4924	0.4924	0.7850
	0.7048	0.6694	0.5215	0.0480	0.0590	0.1101	0.5899	0.7082	0.5195	0.3958
	0.3464	0.7031	0.8103	0.3377	0.7756	0.7780	0.4570	0.3872	0.3536	0.3872
	0.7300	0.4987	0.6760	0.6916	0.7598	0.7896	0.8026	0.8515	0.8138	0.8266
	0.3253	0.6819	0.6893	0.2867	0.1401	0.1592	0.1605	0.3443	0.5930	0.2836
	0.2668	0.5604	0.7434	0.7283	0.6364	0.4873	0.6372	0.5141	0.5141	0.5141
	0.1202	0.0405	0.5887	0.3671	0.2203	0.7526	0.7715	0.7486	0.5102	0.5102
	0.7028	0.1774	0.8189	0.4121	0.2998	0.2224	0.4474	0.3977	0.2626	0.8622
	0.3803	0.7557	0.6283	0.5188	0.3871	0.3099	0.6904	0.7632	0.7065	0.6521
	0.5434	0.5441	0.2832	0.1920	0.2263	0.2108	0.3794	0.4060	0.5116	0.7121
	0.0312	0.3843	0.7871	0.6954	0.2964	0.7305	0.7725	0.7517	0.4978	0.4978
	0.2973	0.4295	-0.0487	0.5682	0.6098	0.7527	0.7450	0.2903	0.2540	0.1597
	0.1268	0.0980	0.3971	0.0419	0.2804	0.2796	0.2748	0.3318	0.2251	0.3288
	0.6196	0.7889	0.6175	0.4019	0.2038	0.1986	0.0345	0.1920	0.8290	0.8290
	0.2286	0.1834	0.5850	0.5597	0.2137	0.3525	0.5443	0.5589	0.4882	0.5083
	0.5454	0.8198	0.4274	0.5916	0.1645	0.1645	0.7698	0.7797	0.7854	0.7848
	0.6743	0.0754	0.4631	0.5453	0.7448	0.7852	0.7397	0.6488	0.5158	0.6507
	0.4925	0.8011	0.3667	0.2111	-0.0899	0.1319	0.8027	0.8383	0.0269	-0.1923
	0.3041	0.3867	0.1567	0.1844	0.7016	0.3118	0.6479	0.6698	0.5171	0.5171
	0.1944	0.5373	0.8110	0.7210	0.8344	0.8416	0.8160	0.5468	0.5889	0.5468
	0.6525	0.3764	0.5919	0.2231	0.5084	0.5465	0.0660	1.0000	1.0000	1.0000
	0.7175	0.6045	0.6624	0.1700	0.8226	0.8088	0.8274	0.8216	0.8278	0.8278
	0.6189	0.8241	0.6692	0.6250	0.2077	0.2262	0.2262	0.4735	0.5919	0.6550
	0.6503	0.7123	0.3988	0.5959	0.7197	0.6603	0.3519	0.0817	0.3819	-0.0369
	0.5268	0.4651	0.0270	0.5681	0.7668	0.8011	0.7235	0.8133	0.8586	0.7024
	0.1800	0.1822	0.4550	0.0670	0.3445	0.3209	0.3137	0.6439	0.1160	0.4880
	0.5004	0.5692	-0.0427	0.4220	0.3045	0.4863	-0.0042	0.1683	0.7010	0.7010
	0.2229	-0.0279	0.4731	0.7688	0.6448	0.4242	0.7269	0.6825	0.4267	0.4267
	-0.1683	0.1389	0.7046	0.4078	0.4798	0.5548	0.0977	0.1421	0.3242	0.2217
	0.6559	0.7905	0.3893	0.3865	0.0395	0.3254	0.3254	0.7151	0.5191	0.7301
	0.6435	0.6084	0.1258	0.3559	0.5243	0.6049	0.5589	0.7122	0.6824	0.5894
	0.4803	0.7987	0.6455	0.4145	0.2597	0.2597	0.5353	0.5757	0.5230	0.5757
	0.7026	0.0288	0.8247	0.5888	0.6074	0.5604	0.5162	0.5748	0.2447	0.6996
	0.6517	0.6000	0.4656	0.6202	0.6265	0.4417	0.2819	0.5500	0.2567	0.7033
	0.5479	0.8046	0.7426	0.8340	0.4362	0.7980	0.7967	0.8386	0.8073	0.8073
	0.4328	0.2416	0.6462	0.8464	0.7940	0.7717	0.5166	0.4667	0.4091	0.4091
	0.2011	0.4038	0.2436	0.4497	0.7835	0.6011	0.6173	0.1969	0.0710	0.3537
	0.7042	0.2203	0.7981	0.7503	0.5953	0.5283	0.7696	0.7338	0.7170	0.2832
	0.2220	0.6362	0.6997	0.5117	0.5395	0.6064	0.1840	0.3564	0.7481	0.7481
	0.5516	0.8222	0.4172	0.2645	0.1536	0.4120	0.1313	-0.0813	-0.0380	0.0700
	-0.0078	0.2778	0.6130	0.7843	0.3970	0.2846	0.8011	0.7995	0.7002	0.7002
	0.4259	0.7399	0.6899	0.6648	0.6568	0.7927	0.8088	0.7987	0.7712	0.7712
	0.3871	0.5214	0.3723	0.4501	0.8437	0.8368	0.7122	0.7518	0.6662	0.7614
	0.7432	0.1950	0.3597	0.8053	0.3505	0.3115	0.3352	0.4195	0.4195	0.4195
	0.6707	0.7661	0.7956	0.0939	0.5710	0.6164	0.6164	0.4676	0.4469	0.3149
	0.5208	0.7840	0.6821	0.8216	0.6570	0.8384	0.7308	0.7654	0.6853	0.6853
	0.6344	0.5221	0.5796	0.5616	0.6081	0.5817	0.4640	0.2016	0.7443	0.7756
	0.5795	0.7883	0.2033	0.5315	0.5575	0.5575	0.8551	0.8566	0.8543	0.8433
	0.6892	0.5239	0.7127	0.7933	0.8553	0.8306	0.2068	0.6439	-0.1019	0.2211
	0.1122	0.3827	0.3264	0.3126	0.3402	0.0796	0.2496	0.6320	0.7609	0.5700
	0.3263	0.6341	0.7621	0.6143	0.7431	0.7431	0.6086	0.6685	0.6685	0.6685
Average	0.4703	0.5148	0.5120	0.4977	0.4978	0.5072	0.5314	0.5330	0.5340	0.5438
Median	0.5238	0.5427	0.5718	0.5153	0.5543	0.5489	0.5553	0.5545	0.5883	0.5750
SD	0.2247	0.2392	0.2448	0.2258	0.2287	0.2298	0.2309	0.2189	0.2347	0.2393
Min	-0.1683	-0.0279	-0.0487	0.0075	-0.0899	0.0637	-0.0745	-0.0813	-0.1019	-0.1923

Max	0.7549	0.8241	0.8326	0.8662	0.8553	0.8416	0.8551	1.0000	1.0000	1.0000
------------	--------	--------	--------	--------	--------	--------	--------	--------	--------	--------

Table A.13: Silhouette tests; constant parts, variable fixtures (Sample 3)

	2f	4f	6f	8f	10f	12f	14f	16f	18f	20f
	0.5222	0.6206	0.4840	0.4785	0.7379	0.8438	0.8398	0.7634	0.7553	0.4769
	0.7255	0.2686	0.7264	0.4599	0.3730	0.6064	0.5442	0.1197	0.0739	0.1164
	0.7566	0.3210	0.7798	0.7003	0.6559	0.3236	0.6670	0.7249	0.7226	0.7317
	0.3753	0.4424	0.6689	0.1047	0.6325	0.3005	0.1561	0.2467	0.2467	0.2496
	0.6857	0.5111	0.5933	0.7114	0.7558	0.6488	0.7198	0.6610	0.6201	0.8477
	0.7129	0.7377	0.1773	0.5730	0.7595	0.4508	0.5776	0.7974	0.5871	0.6032
	0.3385	0.3729	0.3635	0.7584	0.6383	0.7545	0.7877	0.5808	0.5808	0.6417
	0.6977	0.7968	0.0825	0.7181	0.5833	-0.1065	0.2232	0.6749	0.2498	0.4172
	0.0507	0.5266	0.6332	0.6562	0.7290	0.5892	0.5892	0.7844	0.7844	0.7844
	0.3847	0.7091	0.5499	0.2879	0.1388	0.3438	0.4894	0.7380	0.6065	0.6065
	0.2656	0.2252	0.2965	0.1941	0.2866	0.4666	0.4666	0.2217	0.2217	0.2425
	0.5463	0.7394	0.7652	0.4981	0.6816	0.5502	-0.0148	0.3850	0.3850	0.7235
	0.3569	0.0358	0.1925	0.7758	0.6369	0.3410	0.2678	0.4376	0.4376	0.4139
	0.0311	0.3613	0.4594	0.4237	0.5119	0.2032	0.2032	0.5989	0.6460	0.5780
	0.3301	0.3393	0.6425	0.4527	0.5558	0.6265	0.6458	0.1433	0.1433	0.3105
	0.5516	0.7282	0.6895	0.6394	0.5568	0.5779	0.7932	0.6593	0.5405	0.6837
	0.4755	0.5922	0.6072	0.7848	0.8077	0.6822	0.6420	0.5934	0.6214	0.6327
	0.6367	0.7876	0.4675	0.3594	0.3059	0.4361	0.5089	0.6054	0.5763	0.6618
	0.1236	0.1805	0.1158	0.8032	0.4155	0.7327	0.3844	0.8013	0.8013	0.5979
	0.4284	0.8221	0.7149	0.7152	0.5317	0.5317	0.5291	0.6186	0.6186	0.6186
	0.7358	0.0428	0.7523	0.5049	0.3632	0.7396	0.7876	0.2198	0.2198	0.3390
	0.7412	0.2639	0.6694	0.4838	0.1606	-0.1002	-0.0335	0.2355	0.2355	0.5457
	0.2359	0.4500	0.3234	0.4131	0.4445	0.4284	0.4349	0.7899	0.7801	0.7772
	0.2341	0.1408	0.0496	0.2320	0.1396	0.1460	0.6593	0.7881	0.7881	0.7077
	0.6363	0.7856	0.5421	0.2061	0.3086	0.3413	0.3020	0.3372	0.2014	0.6128
	0.2266	0.6432	0.6419	0.4371	0.7108	0.6324	0.3576	0.4966	0.4966	0.3576
	0.6169	0.7999	0.7216	0.6597	0.5721	0.0151	0.1507	0.2042	0.3004	0.7287
	0.6861	0.8289	0.4423	0.0419	0.0311	0.5608	0.6009	0.5804	0.3810	0.5379
	0.5777	0.6577	0.0133	0.3206	0.4003	0.5814	0.7564	0.1274	0.4539	0.4539
	0.0020	0.2711	0.2333	0.2713	0.4592	0.4672	0.1769	0.2365	0.2365	0.3077
	0.6853	0.5730	0.0796	-0.0318	0.5183	0.5999	0.5839	0.6918	0.6918	0.2873
	0.6661	0.6441	0.4458	0.6670	0.3484	0.1431	0.0948	0.7607	0.8911	0.8887
	0.5820	0.7809	0.7236	0.7330	0.6007	0.7295	0.8521	0.6889	0.6114	0.5031
	0.5277	0.6975	0.7548	0.8403	0.8239	0.6613	0.6076	0.5624	0.6421	-0.0564
	0.2666	0.7147	0.7132	0.4366	0.7502	0.4990	0.6425	0.3772	0.3772	0.6425
	-0.2120	0.1797	0.2037	0.8091	0.4003	0.5221	0.5979	0.2878	0.2878	0.3953
	0.6253	0.6278	0.2128	0.3932	0.7077	0.7781	0.7774	0.8282	0.8141	0.5001
	0.6287	0.6737	0.3847	0.6372	0.3999	0.4663	0.8052	0.6985	0.6985	0.8052
	0.7441	0.4459	0.7289	0.3970	0.6363	0.7906	0.7809	0.1697	0.6041	0.6852
	0.7380	0.7491	0.2691	0.6735	0.6951	0.7443	0.6114	0.2837	0.3311	0.3592
	0.1315	0.6253	0.5367	0.7283	0.8209	0.2600	0.5562	0.3501	0.3501	0.3376
	0.3336	0.3364	0.5621	0.4633	0.2621	0.3800	0.1474	0.2758	0.2758	0.1596
	0.6810	0.4795	0.0162	0.4030	0.1542	0.2658	0.7538	0.5180	0.5180	0.7334
	0.6407	0.6529	0.3141	0.4851	0.7697	0.7396	0.7815	0.3903	0.3591	0.7041
	0.5086	0.0583	0.3505	0.2953	0.7374	0.7367	0.4398	0.3234	0.3302	0.3538
	0.4129	0.5070	0.6096	0.7178	0.6947	0.7212	0.7212	0.7012	0.7820	0.1092
	0.4502	0.8213	0.7924	0.1974	0.3470	0.3992	0.2030	0.5049	0.5049	0.2030
	0.5039	0.6878	0.8090	0.4090	0.8286	0.3870	0.5212	0.7602	0.7602	0.4663
	0.7198	0.2607	0.7525	0.7994	0.5083	0.2006	0.3273	0.5557	0.5731	0.4730
	0.7081	0.7991	0.0502	0.3518	0.4624	0.8411	0.8271	0.7637	0.8055	0.6722
	0.7290	0.6679	0.4563	0.1807	0.7249	0.6165	0.7325	0.5683	0.5546	0.3502
	0.0877	0.2224	0.1125	0.7376	0.2134	0.7567	0.6758	0.6704	0.6704	0.7094
	0.7062	0.6152	0.3873	0.2811	0.7413	0.6022	0.6186	0.6165	0.5866	0.5109
	0.6452	0.7635	0.1878	0.5085	0.7677	0.7120	0.5645	0.1431	0.5738	0.5600
	0.4222	0.5264	0.5040	0.3615	0.3427	0.5761	0.5911	0.5150	0.5150	0.4538
	0.4135	0.8240	0.7630	0.6952	0.5814	0.5519	0.4158	0.2182	0.2182	0.2182
	0.7481	0.6570	0.4887	0.7344	0.5861	0.4468	0.1169	0.1249	0.0927	0.7812
	0.6465	0.4498	0.2834	0.3608	0.5146	0.3392	0.2536	0.6998	0.6998	0.1330

	0.6346	0.7581	0.1248	0.4688	0.5327	0.3915	-0.0269	0.4239	0.7858	0.7858
	0.7321	0.2966	0.7425	0.5551	0.2355	0.3700	0.2938	0.4343	0.6440	0.5057
	0.5512	0.6952	0.5405	0.3838	0.2462	0.3895	0.7355	0.5635	0.4886	0.4481
	0.2595	0.1272	0.1062	0.3505	0.3668	0.6028	0.6028	0.3231	0.6332	0.4481
	0.0473	0.5265	0.3256	0.7834	0.7132	0.5830	-0.0986	0.5792	0.5792	0.2598
	0.6870	0.3766	0.6225	0.7385	0.0635	0.2636	0.5171	0.8064	0.6071	0.6523
	0.6443	0.7781	0.3613	0.7591	0.1713	0.3078	0.4523	0.1208	0.5045	0.4729
	0.4915	0.8191	0.6719	0.5419	0.2707	0.8039	-0.0257	0.7710	0.7710	0.6382
	0.3912	0.8226	0.8004	0.8057	0.7247	0.7247	0.7310	0.5074	0.5074	0.5074
	0.2283	0.6590	0.6903	0.7372	0.7349	0.6862	0.6862	0.2543	0.2543	0.2543
	0.7176	0.3802	0.6787	0.7099	0.6440	0.2934	0.4593	0.4724	0.4322	-0.0329
	0.3669	0.5704	0.2668	0.3083	0.6258	0.6596	0.6867	0.7061	0.6279	0.6279
	0.2555	0.1216	0.1674	0.3450	0.2954	0.5995	0.5995	0.3786	0.3940	0.3682
	0.6547	0.3914	0.1608	0.3101	0.6835	0.6911	0.4635	0.7371	0.7371	0.5897
	0.4413	0.5564	0.6408	0.7976	0.8071	0.8451	0.8253	0.8098	0.4721	0.6036
	0.2814	0.2023	0.5457	0.6363	0.2539	0.7119	0.6053	0.5984	0.5984	0.5505
	0.2263	0.6983	0.7658	0.8165	0.8473	0.8001	0.8001	0.4911	0.4911	0.4911
	0.4154	0.4899	0.5235	0.7301	0.7626	0.8054	0.7829	0.7550	0.7712	0.7575
	0.6103	0.7467	0.5210	0.7281	0.4781	0.1920	0.6236	0.5428	0.5428	0.6236
	0.5002	0.8418	0.7672	0.5087	-0.0506	0.3420	0.3282	0.1774	0.1774	0.3282
	0.4918	0.6396	0.7083	0.5893	0.3862	0.1150	0.1150	0.1842	0.7076	0.5738
	0.5486	0.7495	0.7898	0.5588	0.7436	0.6065	0.2746	0.6564	0.6564	0.3922
	0.7535	0.5923	0.6687	0.8639	0.8133	0.2571	0.1213	0.5143	0.4477	0.7153
	0.7061	0.5035	0.6806	0.8152	0.6035	0.5145	0.2590	0.6587	0.7263	0.6823
	0.7098	0.4429	0.4433	0.3772	0.4868	0.5986	0.2045	0.4337	0.4337	0.1785
	0.2111	0.6501	0.6822	0.6384	0.5556	0.4613	0.3631	0.4398	0.4398	0.4398
	0.5683	0.8089	0.2807	0.0782	0.4457	0.4457	0.4700	0.6066	0.4911	0.3424
	0.0568	0.3125	0.3903	0.3342	0.4256	0.0767	0.0767	0.3926	0.3530	0.3939
	0.5127	0.8689	0.6295	0.5179	0.1046	0.1046	0.1478	0.7696	0.5922	0.5922
	0.7408	0.0126	0.8225	0.7548	-0.0759	0.3573	0.3071	0.2575	0.2461	0.4709
	0.6127	0.6930	0.1926	0.7290	0.7479	0.3558	0.7311	0.3850	0.3850	0.6779
	0.6636	0.8264	0.5000	0.3163	0.1832	0.6295	0.6958	0.7603	0.6589	0.4577
	0.5465	0.6038	0.4814	0.0580	0.2119	0.6735	0.7970	0.7340	0.7340	0.7970
	0.3538	0.8015	0.7863	0.1492	0.6225	0.3265	0.7803	0.4694	0.4694	0.7556
	0.7135	0.7116	0.2880	0.6431	0.6678	0.6878	0.5353	0.3388	0.4688	0.4730
	0.2934	0.7517	0.6908	0.3423	0.6248	0.4674	0.6439	0.5801	0.5801	0.4884
	0.3854	0.3849	0.1146	0.4811	0.4653	0.3812	0.4246	0.8340	0.8020	0.7398
	0.2093	0.6879	0.7631	0.7805	0.8009	0.7450	0.7213	0.2384	0.2384	0.2384
	0.5031	0.7745	0.5476	0.5741	0.4283	0.7709	0.5565	0.5224	0.5224	0.8376
	0.6944	0.2413	0.6415	0.4727	0.3029	0.2445	0.2721	0.4727	0.3896	0.0052
	0.6652	0.7774	0.2167	0.5942	0.2285	0.1673	0.1246	0.0386	0.2713	0.4312
	0.7230	0.2766	0.6127	-0.0243	0.1833	0.7629	0.5761	0.2357	0.2357	0.4340
Mean	0.4923	0.5501	0.4885	0.5159	0.5019	0.4999	0.4905	0.5020	0.5130	0.5024
Median	0.5464	0.6230	0.5386	0.5086	0.5322	0.5410	0.5563	0.5202	0.5315	0.5044
SD	0.2189	0.2298	0.2349	0.2225	0.2291	0.2228	0.2489	0.2165	0.1946	0.2076
Min	-0.2120	0.0126	0.0133	-0.0318	-0.0759	-0.1065	-0.0986	0.0386	0.0739	-0.0564
Max	0.7566	0.8689	0.8225	0.8639	0.8473	0.8451	0.8521	0.8340	0.8911	0.8887

A.2.1. Silhouette Plots

Figure A.1 to Figure A.30 present the silhouette plots for the problems tested in Table A.11 to Table A.13. The problems sizes and test run repetition are shown in the figure captions.

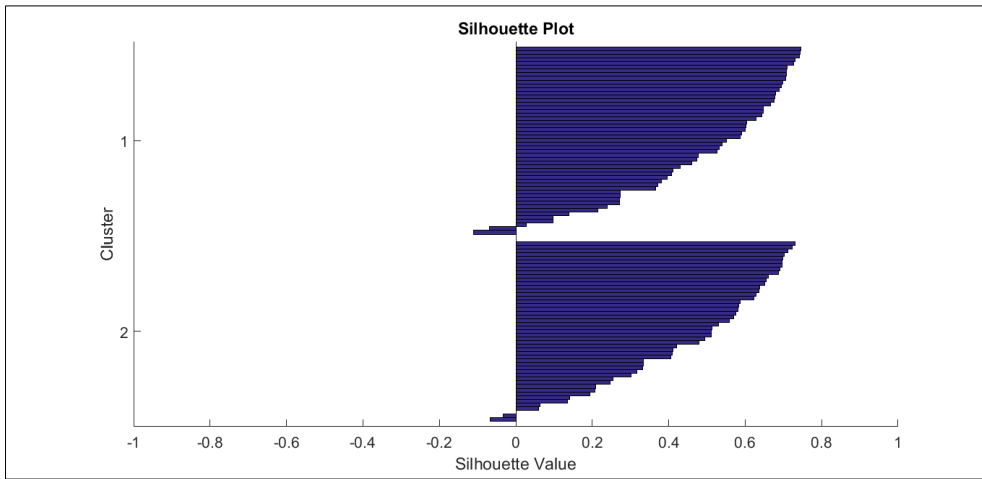


Figure A.1: 100 parts; 2 fixtures (1)

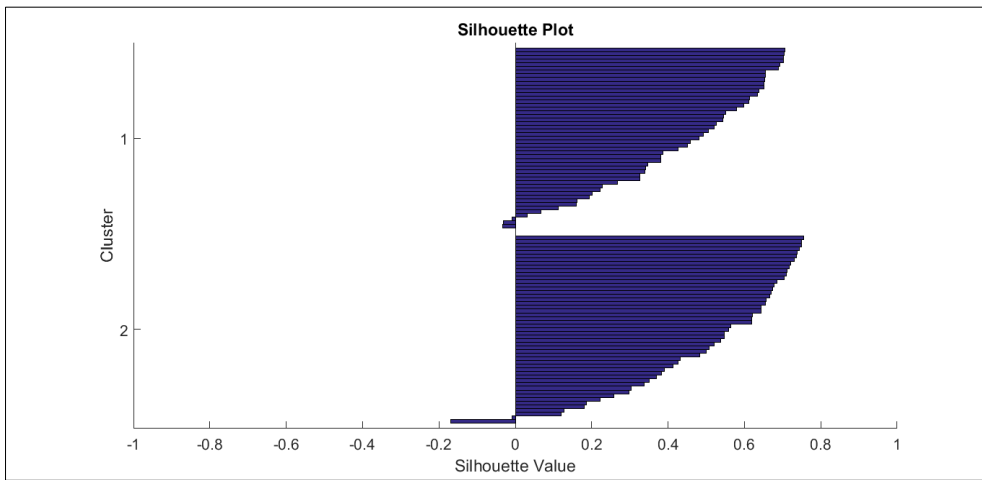


Figure A.2: 100 parts; 2 fixtures (2)

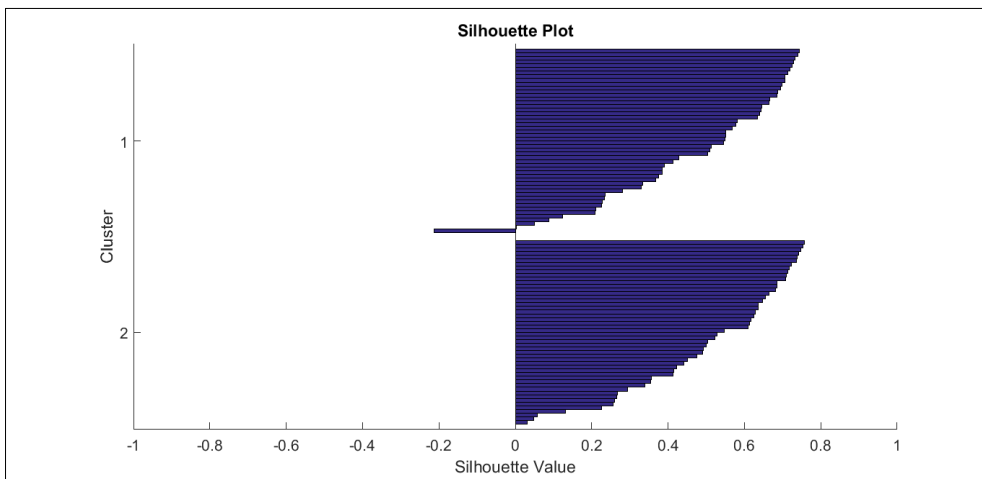


Figure A.3: 100 parts; 2 fixtures (3)

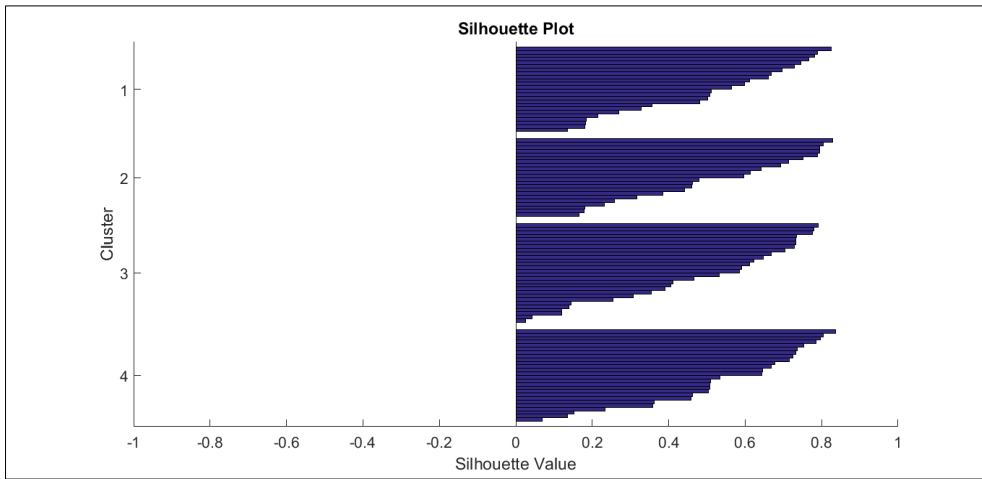


Figure A.4: 100 parts; 4 fixtures (1)

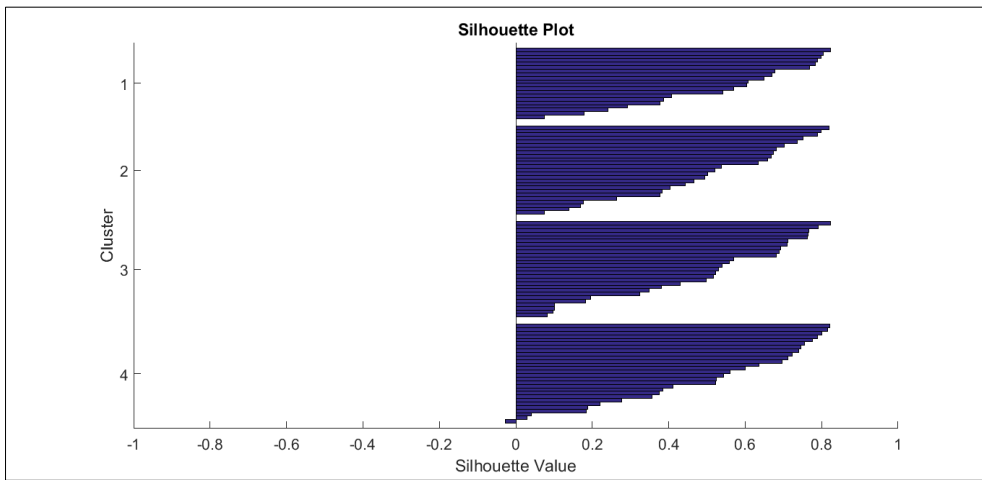


Figure A.5: 100 parts; 4 fixtures (2)

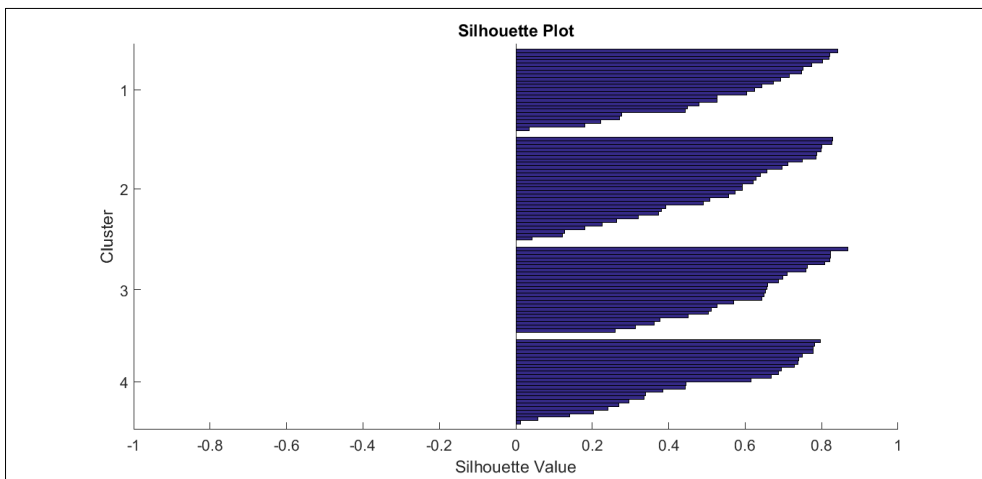


Figure A.6: 100 parts; 4 fixtures (3)

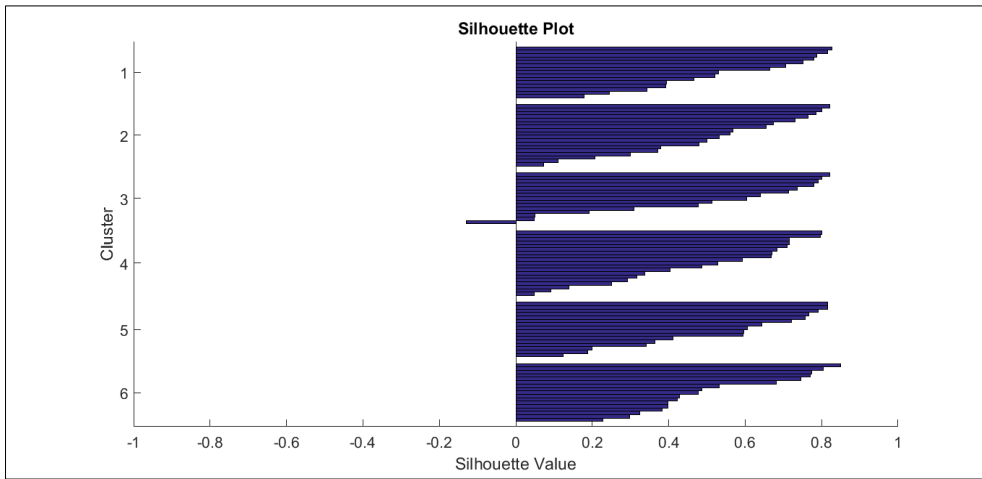


Figure A.7: 100 parts; 6 fixtures (1)

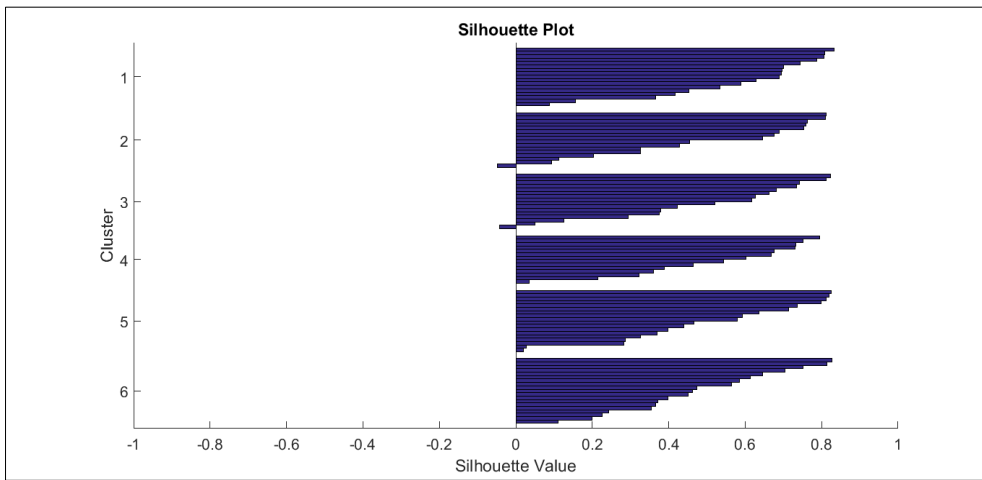


Figure A.8: 100 parts; 6 fixtures (2)

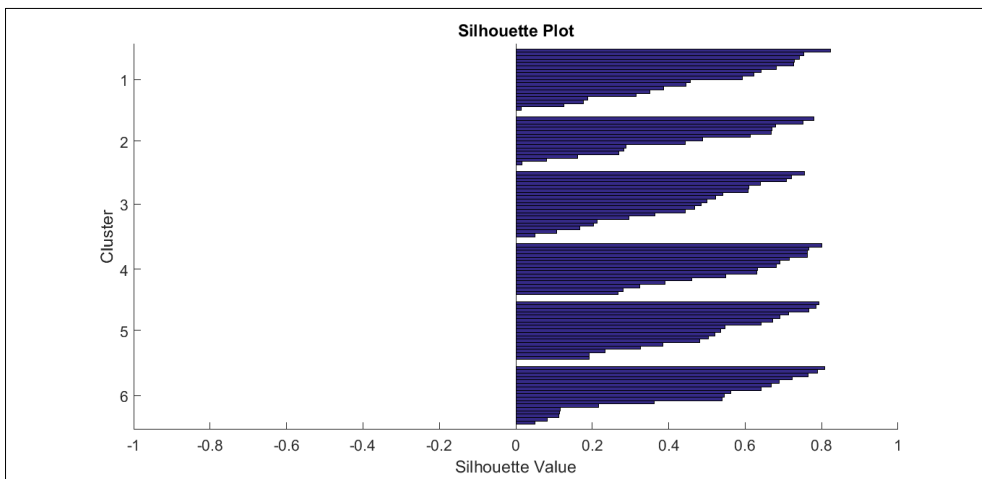


Figure A.9: 100 parts; 6 fixtures (3)

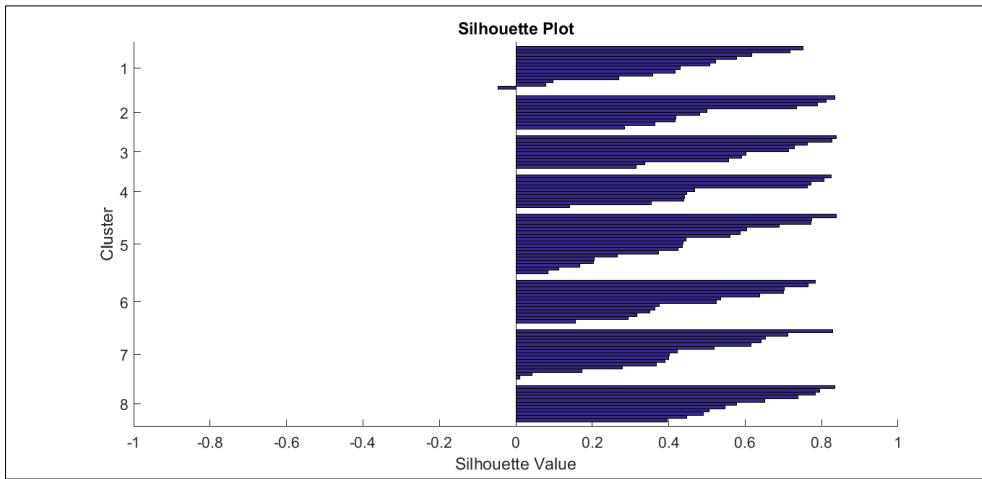


Figure A.10: 100 parts; 8 fixtures (1)

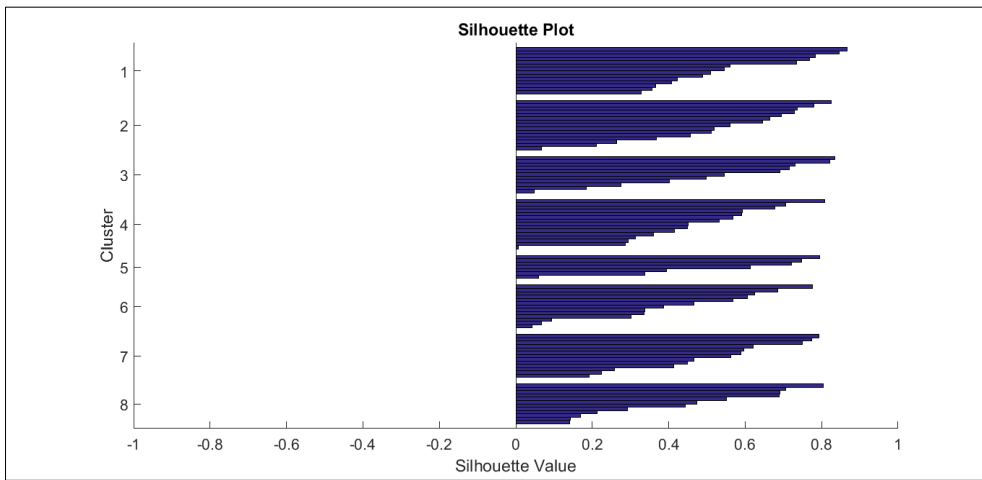


Figure A.11: 100 parts; 8 fixtures (2)

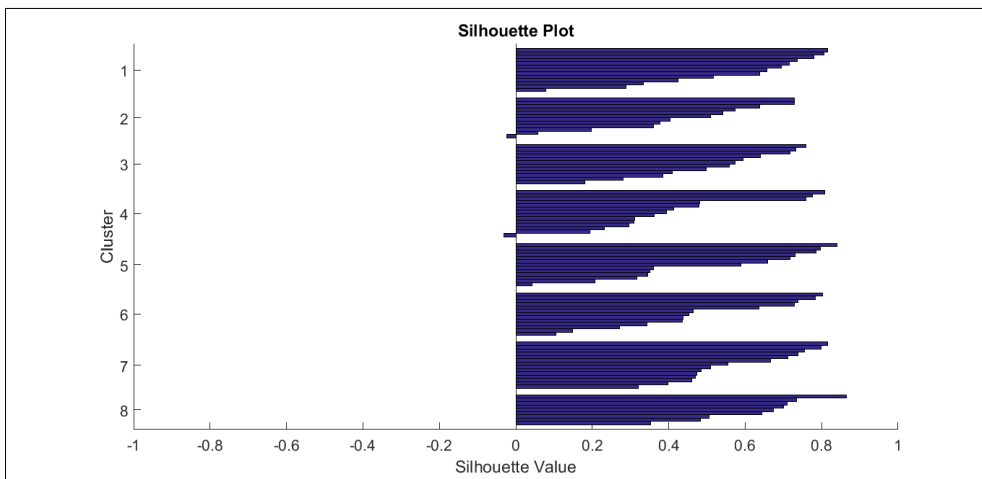


Figure A.12: 100 parts; 8 fixtures (3)

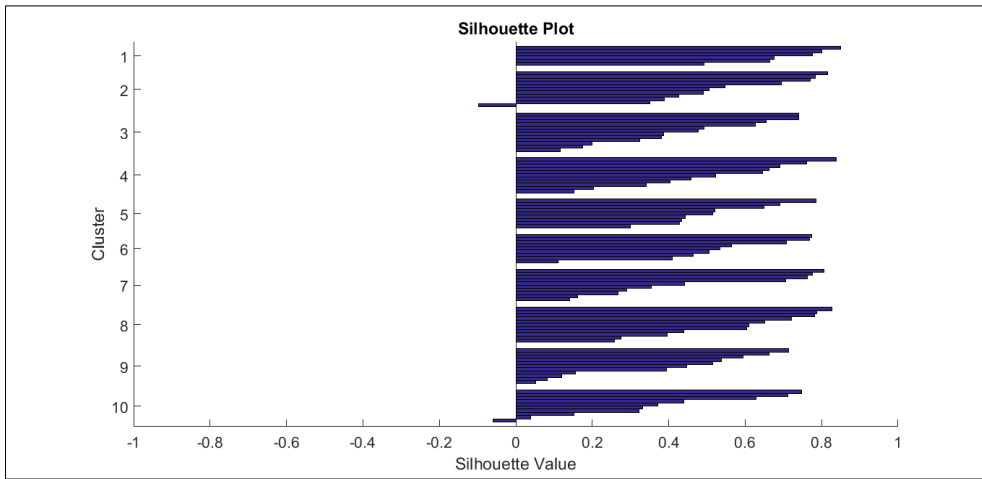


Figure A.13: 100 parts; 10 fixtures (1)

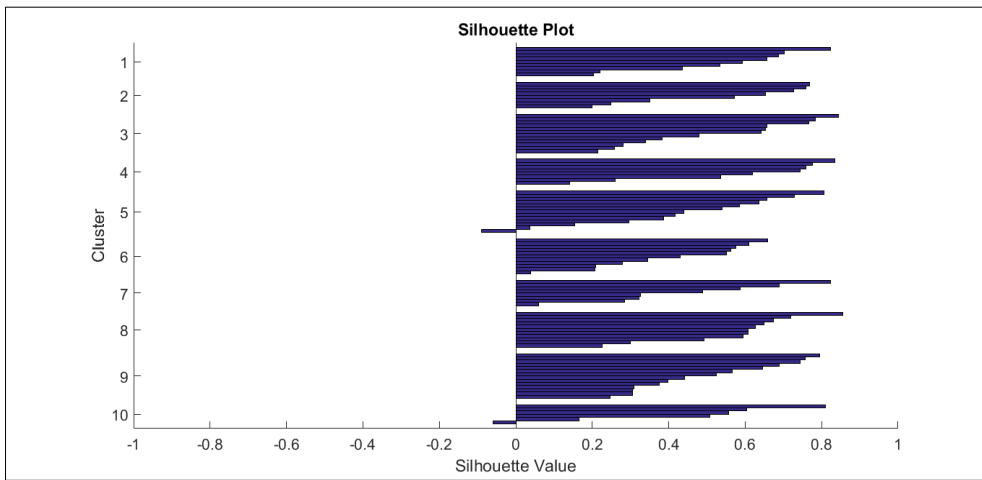


Figure A.14: 100 parts; 10 fixtures (2)

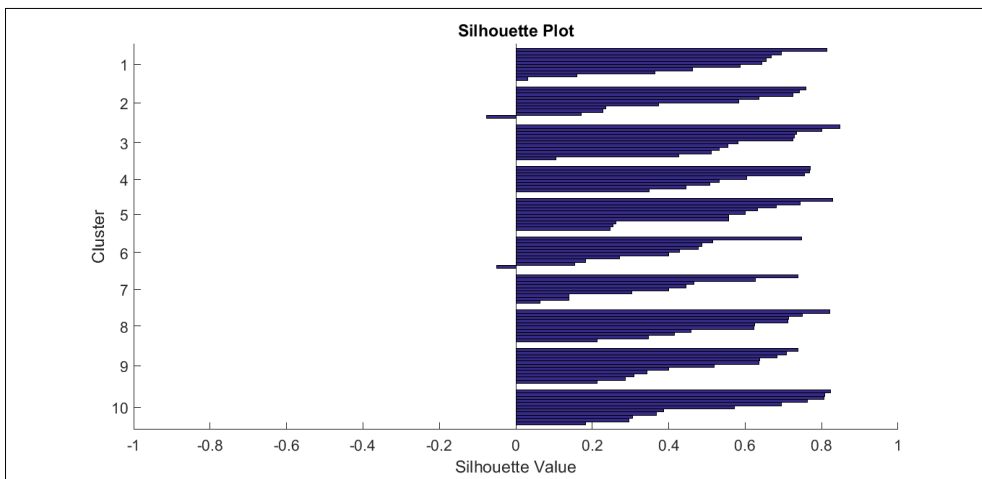


Figure A.15: 100 parts; 10 fixtures (3)

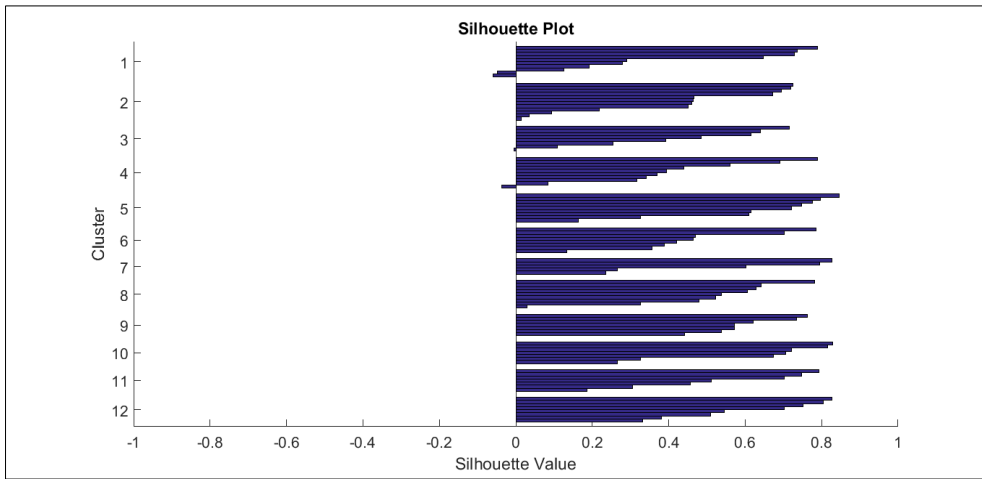


Figure A.16: 100 parts; 12 fixtures (1)

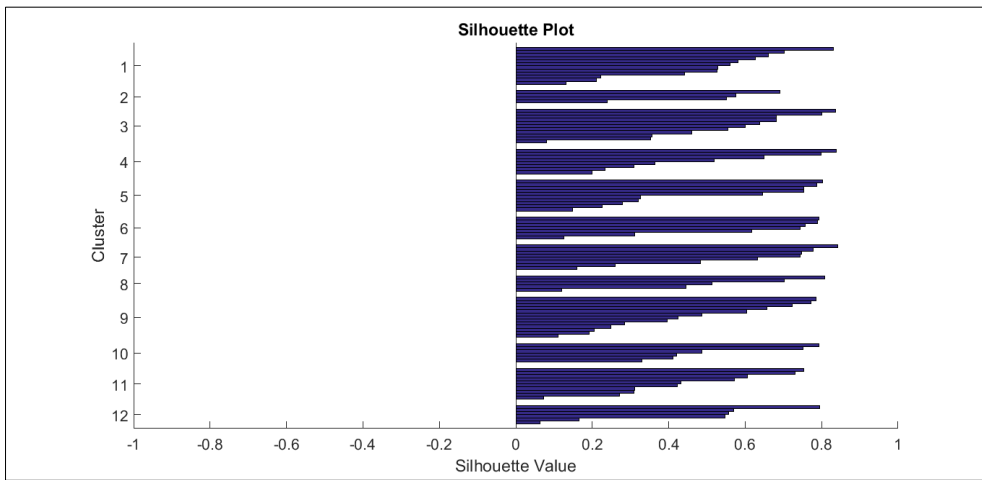


Figure A.17: 100 parts; 12 fixtures (2)

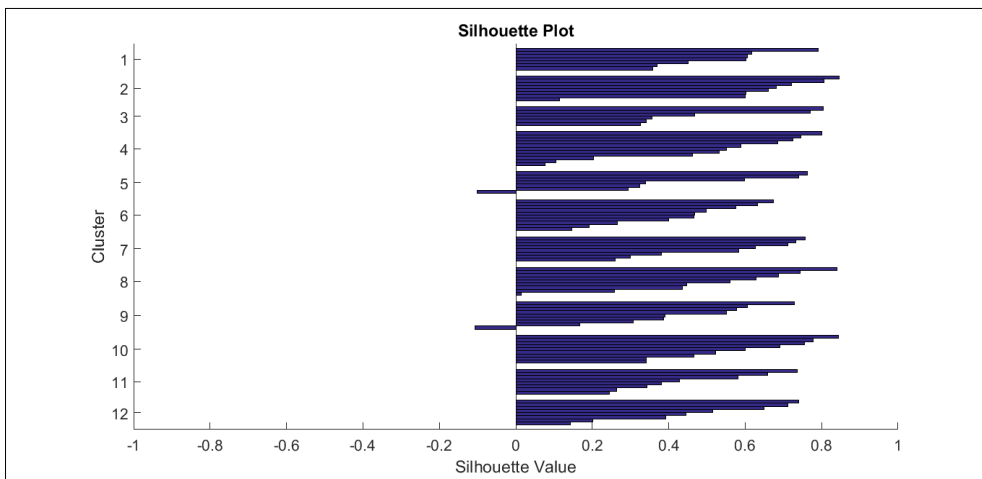


Figure A.18: 100 parts; 12 fixtures (3)

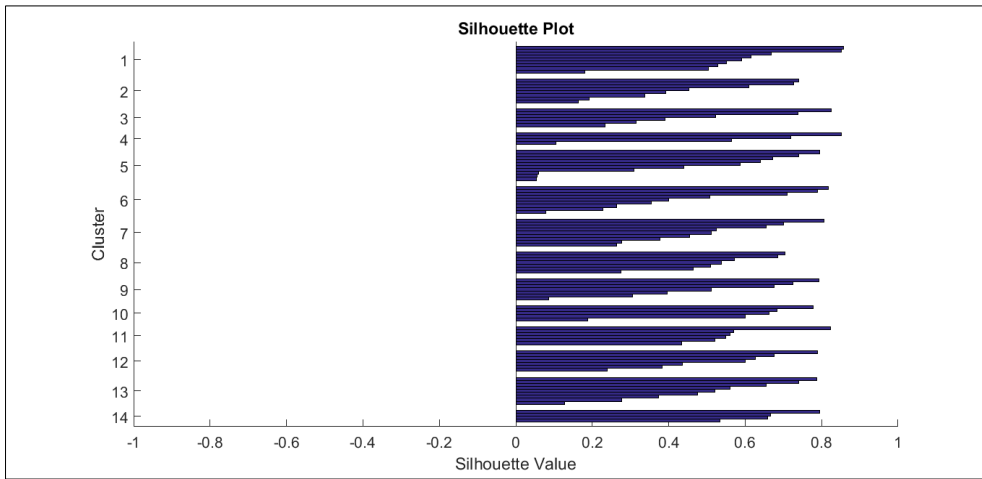


Figure A.19: 100 parts; 14 fixtures (1)

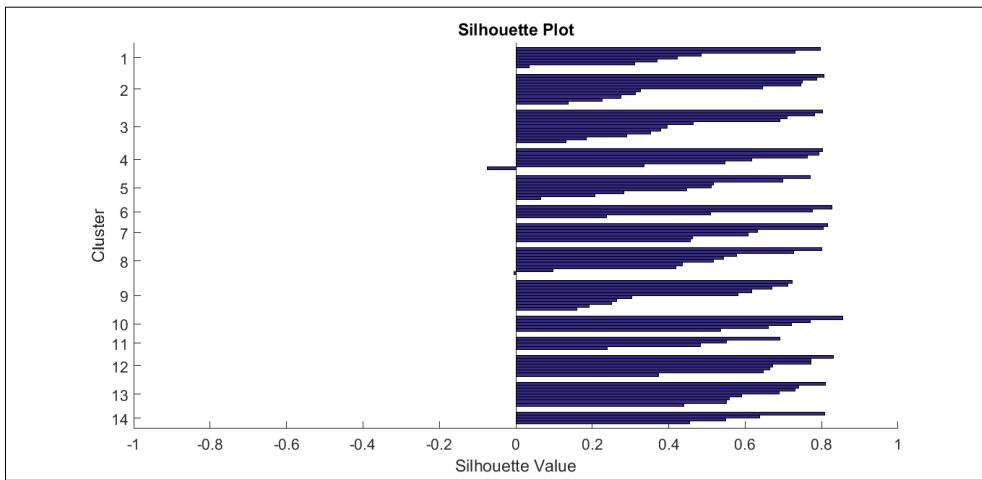


Figure A.20: 100 parts; 14 fixtures (2)

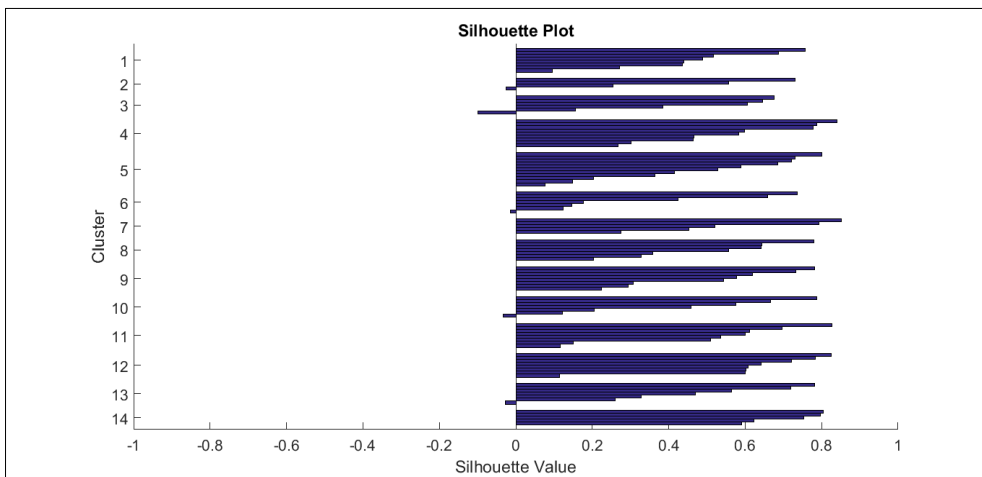


Figure A.21: 100 parts; 14 fixtures (3)

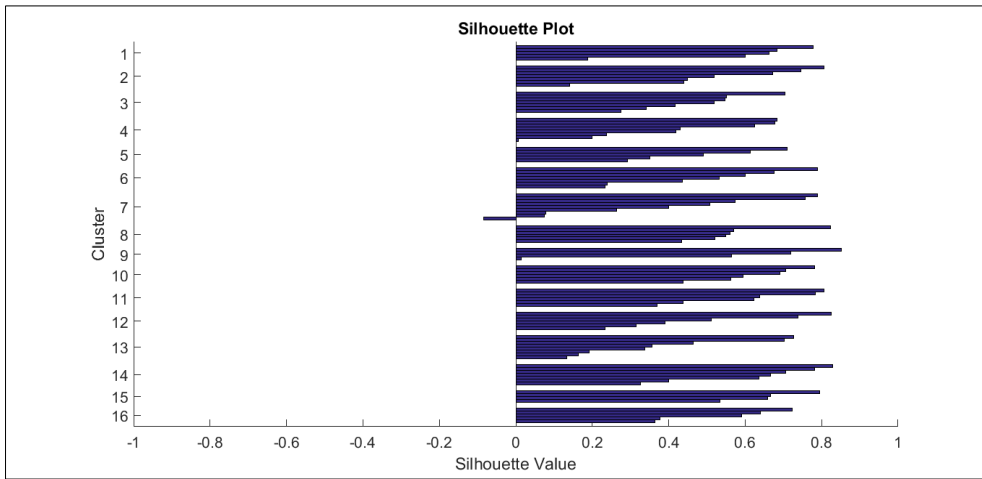


Figure A.22: 100 parts; 16 fixtures (1)

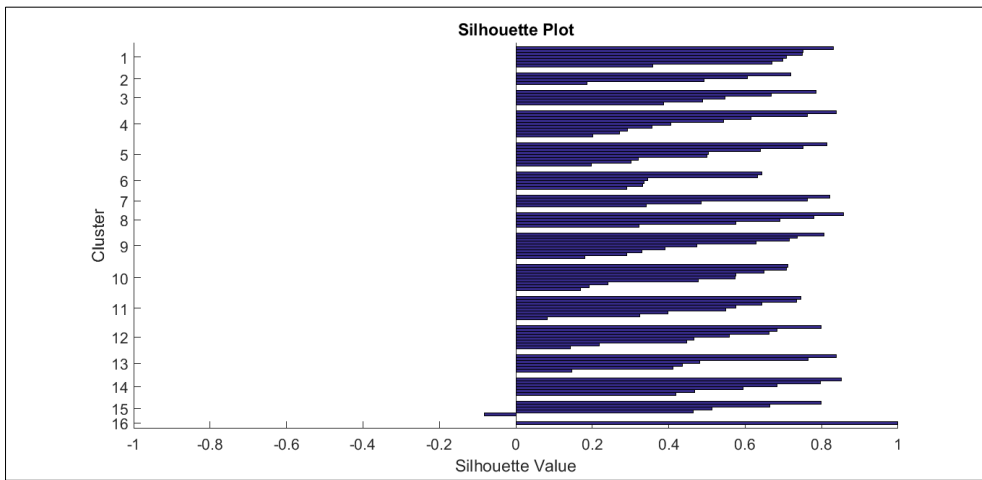


Figure A.23: 100 parts; 16 fixtures (2)

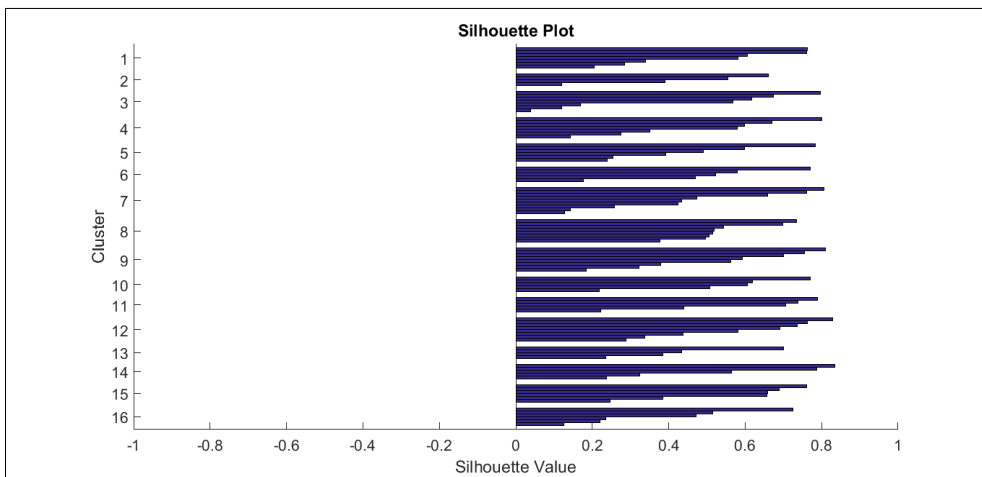


Figure A.24: 100 parts; 16 fixtures (3)

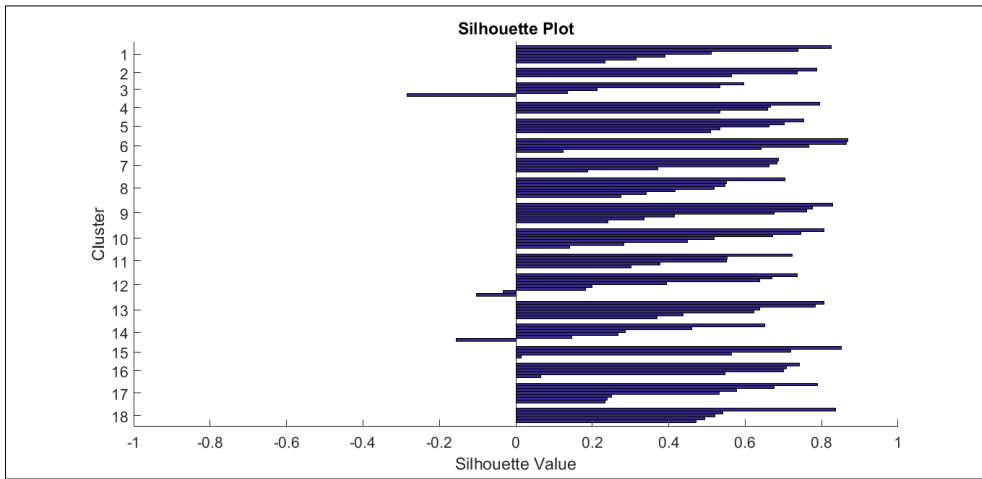


Figure A.25: 100 parts; 18 fixtures (1)

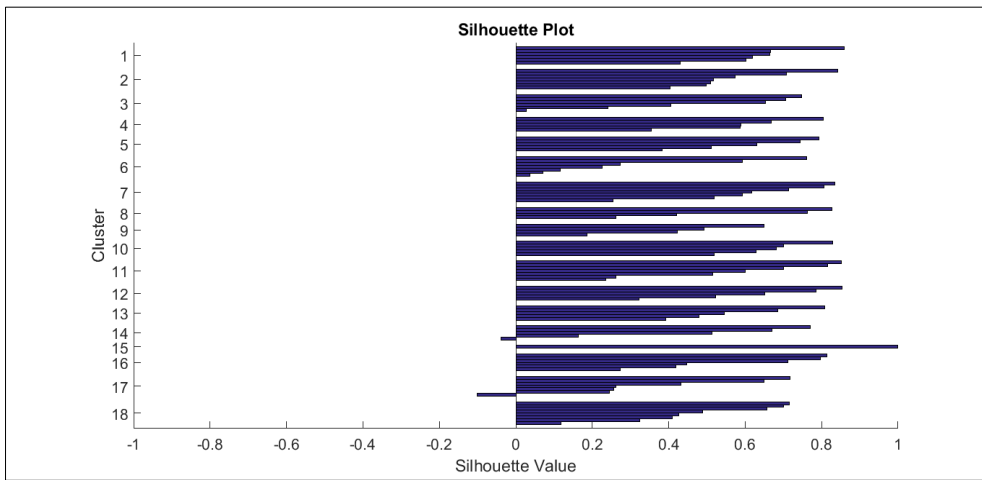


Figure A.26: 100 parts; 18 fixtures (2)

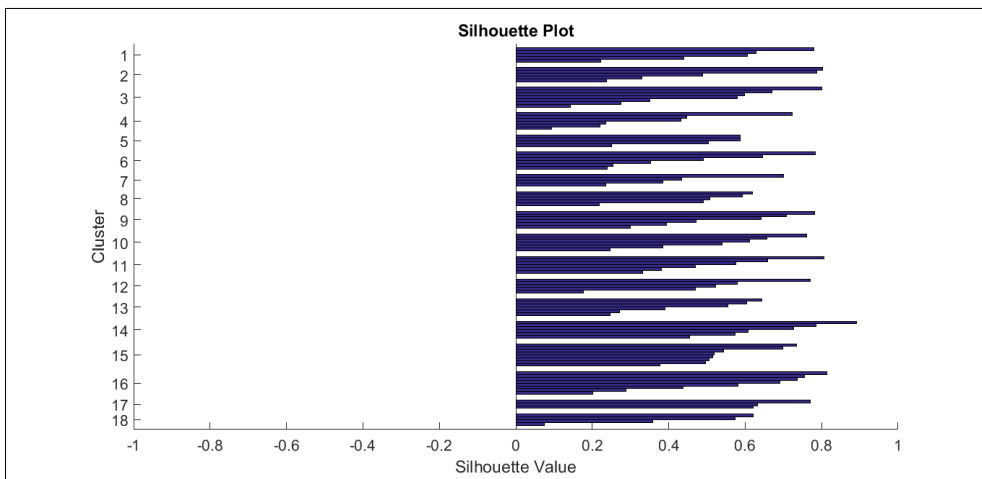


Figure A.27: 100 parts; 18 fixtures (3)

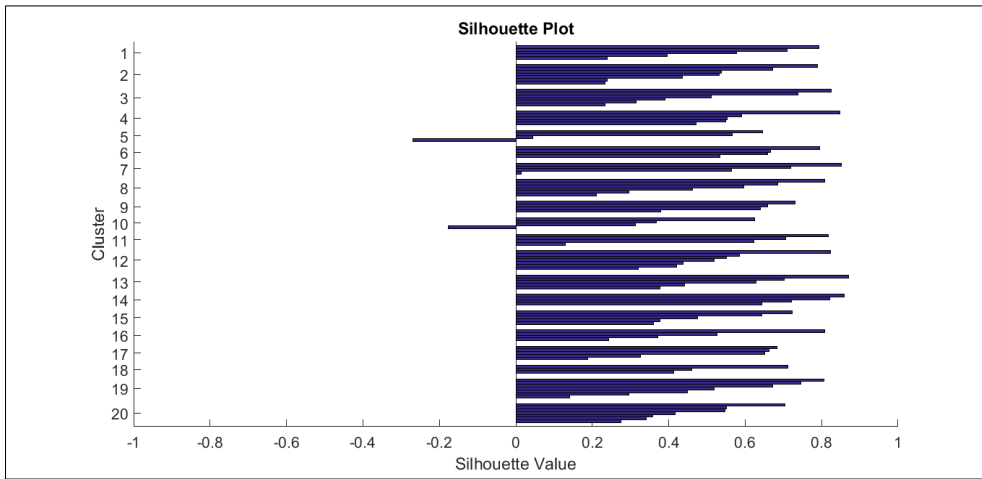


Figure A.28: 100 parts; 20 fixtures (1)

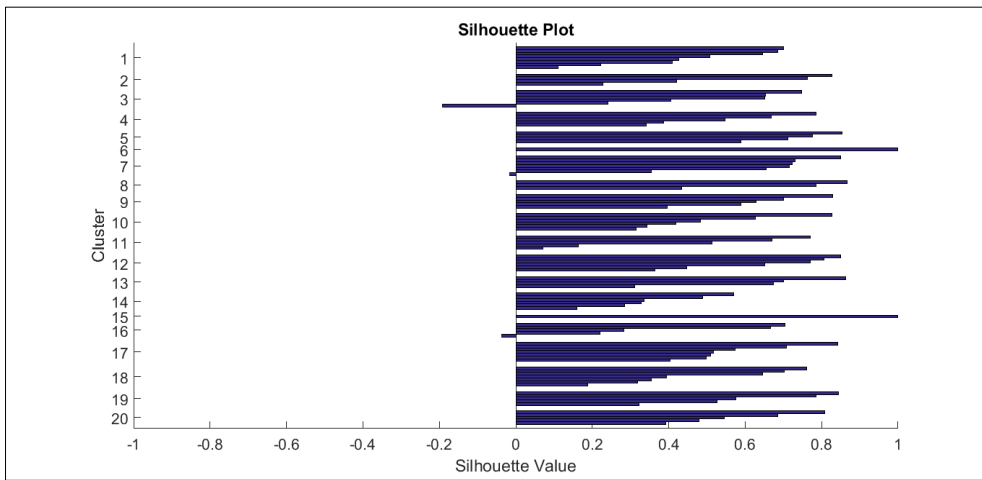


Figure A.29: 100 parts; 20 fixtures (2)

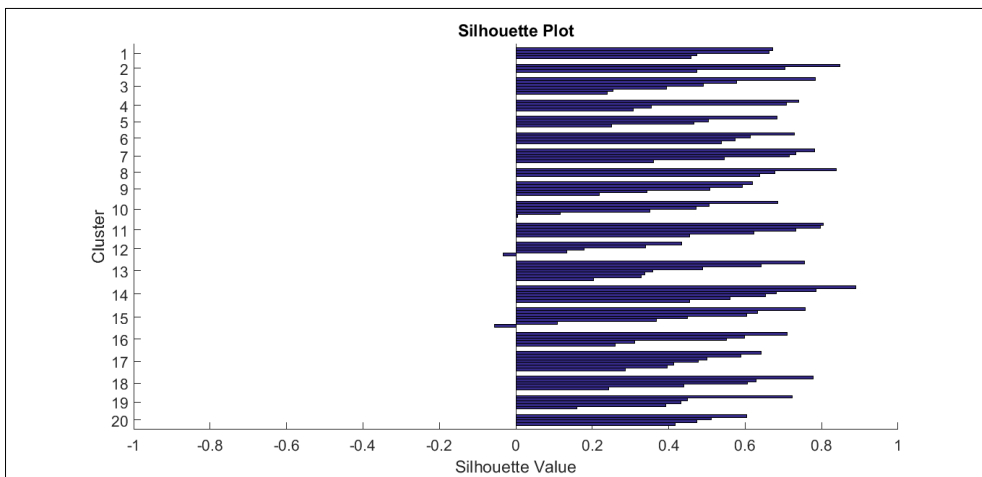


Figure A.30: 100 parts; 20 fixtures (3)

A.3. Cluster Reordering

Table A.14 to Table A.23 present the percentage improvement in total cumulative pairwise dissimilarities for both the optimal order in comparison to the default order (as per default dendrogram), and the optimal order in comparison to the initial order (as per k-means clustering output). The number of fixtures were increased for each test. The test samples are as stated in the table captions.

Table A.14: Percentage improvements (Sample 1)

Fixture Name	Optimal/Default (%)	Optimal/Initial (%)
4 Fixtures		
1	5.556914	10.94799
2	4.534312	13.84305
3	5.584463	12.0162
4	5.71754	9.083077
10 Fixtures		
1	5.722385	8.659912
2	6.322938	11.62347
3	0	3.492571
4	4.80981	10.88428
5	3.989917	4.782372
6	5.564008	10.70564
7	0.301781	7.360116
8	5.348175	15.05876
9	3.586205	8.535588
10	5.09478	7.939859
16 Fixtures		
1	1.68653	9.991953
2	0	3.492571
3	2.707866	15.65894
4	0.611466	2.70825
5	3.134546	5.494025
6	0.823644	7.294115
7	0	0
8	0	10.77055
9	3.723337	7.94156
10	0.071712	9.263901
11	3.822586	9.158656
12	5.473749	14.41645
13	1.266707	10.3715
14	4.677492	1.680046
15	4.124115	4.826261
16	4.979394	10.91751

Table A.15: Percentage improvements (Sample 2)

Fixture Name	Optimal/Default (%)	Optimal/Initial (%)
4 Fixtures		
1	5.37445	9.01537
2	6.001908	10.27248
3	4.268417	9.870894
4	5.06634	11.09251
10 Fixtures		
1	5.491532	5.11051
2	2.017765	5.463029
3	5.352146	5.673275
4	1.745708	9.132963
5	4.543259	3.606903
6	7.391736	15.6828
7	0.958081	12.59472
8	4.433851	9.368852
9	6.052361	8.715658

10	7.588297	9.904856
16 Fixtures		
1	1.686427	9.833872
2	2.746661	3.922306
3	3.491145	6.948091
4	5.750918	6.710415
5	2.852972	7.427527
6	2.228345	7.343536
7	4.937493	4.27266
8	5.808783	3.370088
9	0.591221	17.98603
10	4.008495	7.956349
11	0	1.546417
12	1.451544	7.15276
13	2.295636	1.845527
14	2.586279	1.314199
15	3.436407	8.114377
16	2.251005	5.723204

Table A.16: Percentage improvements (Sample 3)

Fixture Name	Optimal/Default (%)	Optimal/Initial (%)
4 Fixtures		
1	4.060693	11.53239
2	5.130501	12.7251
3	4.695929	8.185784
4	4.771388	11.70535
10 Fixtures		
1	4.269245	8.471521
2	1.952043	3.611028
3	1.847164	8.678812
4	3.751644	10.38644
5	4.62446	7.438031
6	4.110432	11.15509
7	4.806779	17.64918
8	6.160036	3.570492
9	4.614802	12.64016
10	2.71831	4.655604
16 Fixtures		
1	5.295849	3.167375
2	2.738797	0
3	1.977664	10.59384
4	6.215946	4.54607
5	3.486087	8.194001
6	2.717562	4.169753
7	7.821666	9.575515
8	5.524091	6.253523
9	8.005444	16.24132
10	3.466739	8.214307
11	0	0
12	4.904599	7.612392
13	3.338029	10.2669
14	1.166583	8.716027
15	2.71831	4.655604
16	3.740073	10.28976

Table A.17: Percentage improvements (Sample 4)

Fixture Name	Optimal/Default (%)	Optimal/Initial (%)
4 Fixtures		
1	3.882421	12.95866
2	3.474375	10.98434
3	5.322129	13.75595

4	5.505827	12.88221
10 Fixtures		
1	3.572162	6.938447
2	5.495094	14.37817
3	5.575511	11.04173
4	7.155756	8.013373
5	3.678169	12.63849
6	3.371659	10.94039
7	1.485035	10.93878
8	8.459249	13.29285
9	7.386962	12.48359
10	4.268802	12.09176
16 Fixtures		
1	1.214887	8.372371
2	1.973806	9.560698
3	3.81559	10.98678
4	4.680414	8.898684
5	0	4.301619
6	0	0
7	1.461378	2.966011
8	1.94776	5.549
9	2.933329	7.745052
10	1.882366	8.379531
11	3.91684	11.26556
12	6.097277	10.28509
13	4.200593	12.57006
14	1.033902	8.713104
15	1.081922	10.90979
16	6.004378	16.53185

Table A.18: Percentage improvements (Sample 5)

Fixture Name	Optimal/Default (%)	Optimal/Initial (%)
4 Fixtures		
1	3.283171	7.646828
2	5.221648	12.56345
3	4.772935	16.32076
4	6.300134	10.20924
10 Fixtures		
1	3.508415	14.32255
2	2.176415	6.741274
3	4.361851	6.934141
4	4.171897	10.79164
5	2.924623	9.956403
6	5.907701	5.537727
7	2.579998	9.558081
8	2.610436	10.17845
9	7.567193	5.787221
10	1.043625	10.67961
16 Fixtures		
1	4.429658	3.709042
2	0	1.322971
3	0	5.990665
4	0.810673	2.361124
5	0	7.06888
6	3.79868	6.379104
7	2.575684	12.15331
8	1.203993	1.732792
9	0	8.873587
10	2.952483	6.437593
11	4.311992	4.445165
12	3.876798	8.353542
13	7.832939	3.530065

14	7.361998	6.808676
15	5.657481	5.657481
16	0.536358	8.739366

Table A.19: Percentage improvements (Sample 6)

Fixture Name	Optimal/Default (%)	Optimal/Initial (%)
4 Fixtures		
1	5.587233	12.9198
2	7.095851	12.7836
3	7.459653	9.667365
4	6.3841	12.76024
10 Fixtures		
1	5.102831	18.33817
2	0.906228	13.64503
3	1.633114	6.065941
4	2.40698	5.217812
5	5.960453	6.505275
6	3.322038	5.996811
7	1.544046	11.73525
8	5.844568	15.22065
9	10.34329	7.705578
10	6.000624	9.73328
16 Fixtures		
1	1.902936	9.344456
2	2.740899	8.605435
3	0	7.448615
4	3.62169	3.62169
5	3.429528	9.288658
6	4.332517	2.129366
7	3.965176	12.09763
8	4.71197	0
9	5.449585	10.94696
10	6.375455	6.487938
11	1.214599	6.194509
12	1.986274	4.63944
13	1.006591	4.626548
14	0	7.021306
15	4.076834	11.51443
16	7.251138	14.2572

Table A.20: Percentage improvements (Sample 7)

Fixture Name	Optimal/Default (%)	Optimal/Initial (%)
4 Fixtures		
1	5.1385	14.68426
2	5.883876	12.79049
3	4.314786	15.13161
4	6.273409	14.44882
10 Fixtures		
1	4.946762	11.35784
2	4.757157	10.80059
3	5.431914	13.76634
4	3.67243	8.058126
5	2.2612	7.781085
6	5.531677	10.22385
7	8.882646	9.176335
8	7.562037	5.602946
9	7.407185	10.83281
10	4.342867	10.84539
16 Fixtures		
1	0.82451	9.189419
2	6.815958	12.07575

3	1.625865	9.257998
4	1.168341	5.87761
5	2.689397	9.603748
6	3.173653	6.15206
7	7.442772	6.70551
8	5.744142	6.770555
9	6.497051	19.79139
10	4.771516	6.689605
11	0.676516	6.838673
12	3.41912	8.798684
13	0	3.012039
14	5.866424	12.01605
15	3.072296	9.282424
16	2.911103	11.2608

Table A.21: Percentage improvements (Sample 8)

Fixture Name	Optimal/Default (%)	Optimal/Initial (%)
4 Fixtures		
1	7.217497	17.14333
2	7.505413	10.85544
3	6.15384	10.53695
4	4.909904	10.00992
10 Fixtures		
1	4.081934	9.771837
2	3.589846	9.67836
3	4.786899	3.687977
4	5.430117	9.707538
5	3.25237	10.33848
6	2.661925	6.032178
7	5.75677	9.503961
8	1.451679	8.745699
9	7.216494	14.78571
10	6.09816	5.696183
16 Fixtures		
1	2.405376	4.623133
2	0.696782	4.3699
3	1.710037	0
4	2.884341	4.188234
5	3.297997	1.351077
6	0.665938	21.10761
7	1.66E-14	5.578779
8	2.137502	7.524882
9	4.277642	8.030486
10	6.710299	12.66501
11	0	5.79426
12	2.686229	3.40772
13	8.782833	5.224139
14	3.995166	10.8863
15	3.077037	1.309337
16	0	0

Table A.22: Percentage improvements (Sample 9)

Fixture Name	Optimal/Default (%)	Optimal/Initial (%)
4 Fixtures		
1	8.498126	17.01243
2	7.253322	12.63228
3	6.579537	15.39358
4	5.298815	12.30825
10 Fixtures		
1	8.652505	13.9793
2	8.317581	10.28865

3	5.667272	11.05062
4	6.728343	10.91518
5	3.279801	6.54732
6	5.34039	7.427106
7	4.022969	8.606017
8	2.368818	5.345502
9	7.466286	10.15959
10	2.78303	11.04774
16 Fixtures		
1	0	5.429023
2	5.156143	2.347064
3	3.381787	9.013517
4	2.204025	9.879339
5	1.733838	8.102106
6	7.285608	16.98183
7	5.542499	2.68713
8	1.072667	0
9	2.254925	8.382459
10	4.06175	7.395173
11	2.368818	5.345502
12	6.365094	4.075157
13	2.416868	12.64729
14	1.732616	7.927895
15	2.286144	13.74586
16	3.222136	10.71982

Table A.23: Percentage improvements (Sample 10)

Fixture Name	Optimal/Default (%)	Optimal/Initial (%)
4 Fixtures		
1	3.962971	12.12837
2	6.206298	11.31671
3	6.128463	11.15037
4	5.251781	14.39334
10 Fixtures		
1	4.604914	10.69225
2	2.596776	8.073791
3	6.065868	12.18148
4	3.312227	6.338433
5	5.282266	5.419587
6	3.532417	6.555337
7	4.532763	15.57193
8	3.598905	9.308372
9	4.077535	9.318215
10	3.109142	9.140111
16 Fixtures		
1	3.810537	14.95571
2	4.064353	15.70087
3	5.289294	8.475423
4	2.127215	5.267528
5	0	5.305523
6	3.536004	6.67348
7	1.530026	1.530026
8	2.044006	9.079176
9	4.071227	3.728479
10	5.124852	11.95843
11	5.549941	5.552061
12	4.082858	7.767787
13	5.158187	20.09316
14	4.880136	7.297365
15	3.093112	9.892236
16	5.743693	11.17228

A.3.1. Dendrograms

Figure A.31 to Figure A.60 present the default leaf order dendrogram and optimal leaf order dendrogram below it. Each figure represents the intracluster sequence for each fixture, with the problem size and fixture name shown in the figure caption (only first test sample shown, as described in Table A.14).

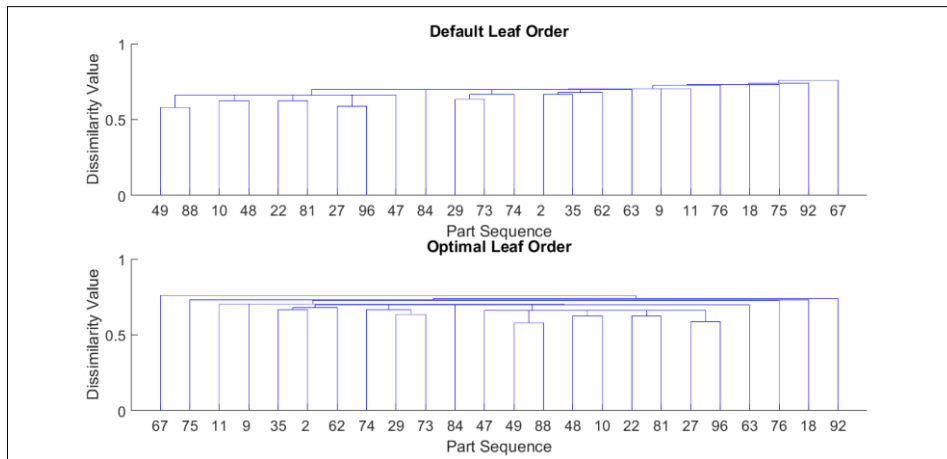


Figure A.31: 100 parts, 4 fixtures (Fixture 1)

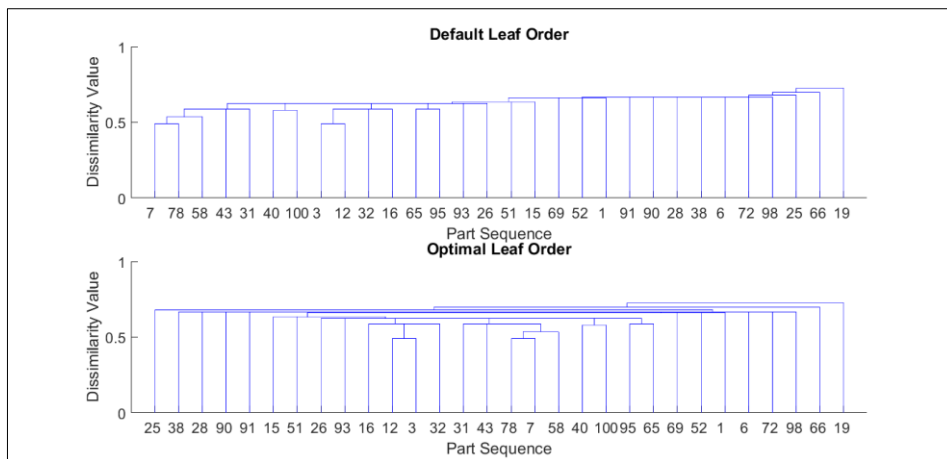


Figure A.32: 100 parts, 4 fixtures (Fixture 2)

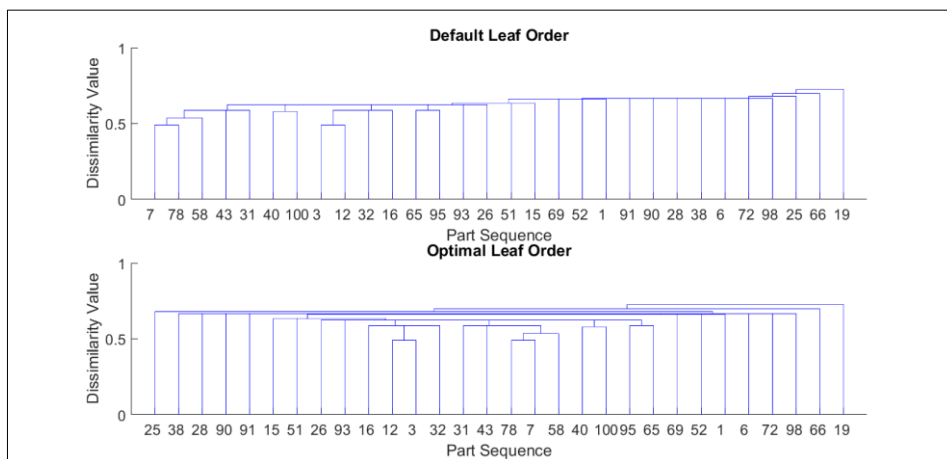


Figure A.33: 100 parts, 4 fixtures (Fixture 3)

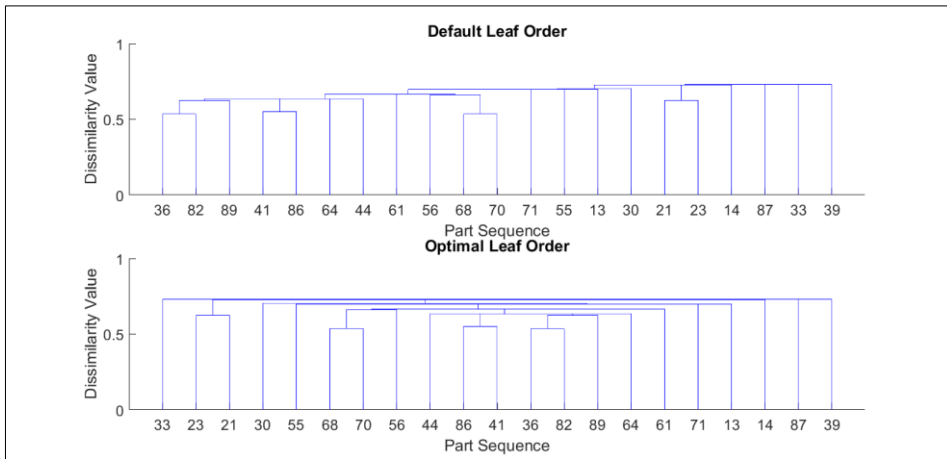


Figure A.34: 100 parts, 4 fixtures (Fixture 4)

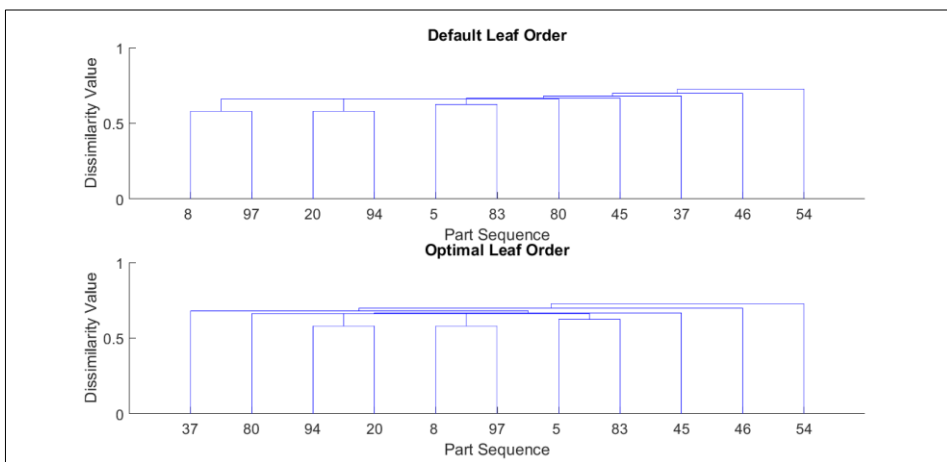


Figure A.35: 100 parts, 10 fixtures (Fixture 1)

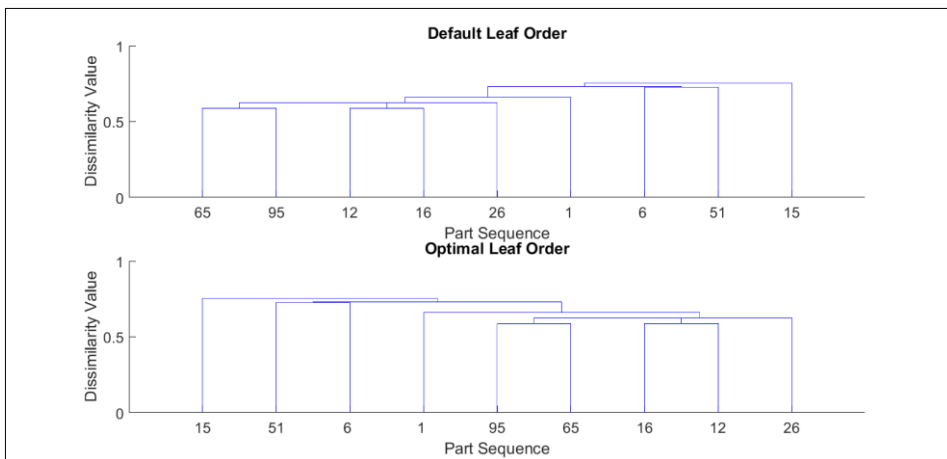


Figure A.36: 100 parts, 10 fixtures (Fixture 2)

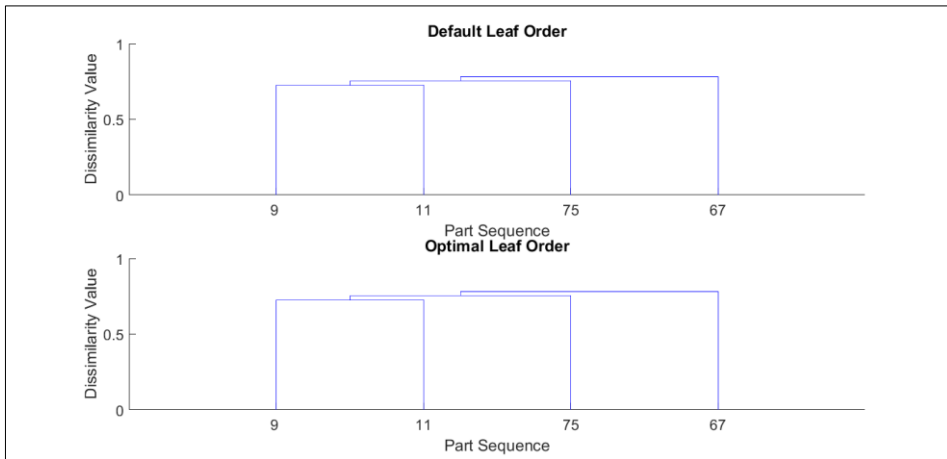


Figure A.37: 100 parts, 10 fixtures (Fixture 3)

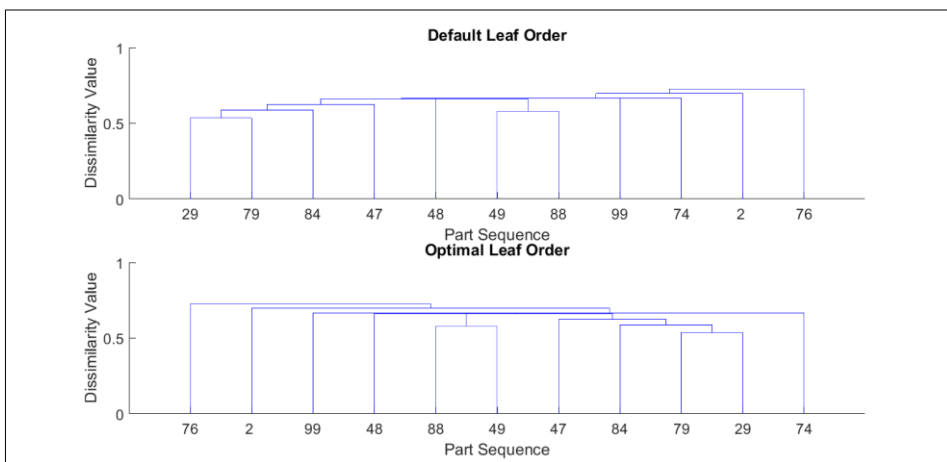


Figure A.38: 100 parts, 10 fixtures (Fixture 4)

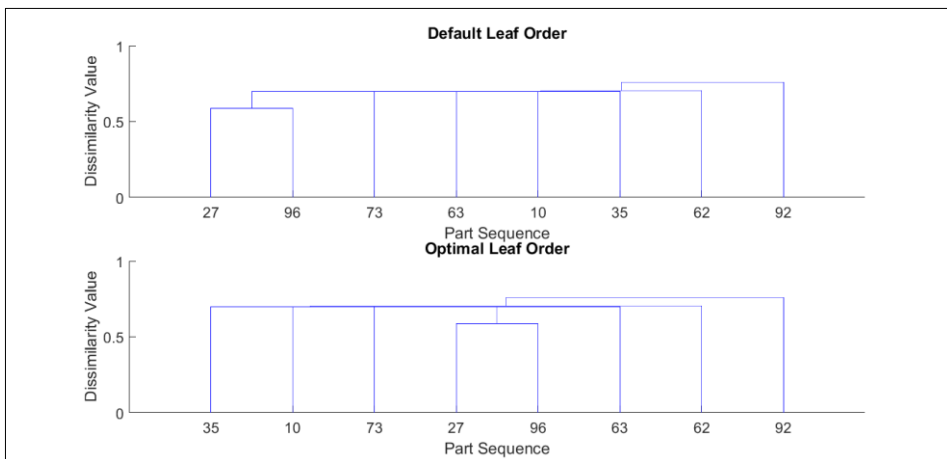


Figure A.39: 100 parts, 10 fixtures (Fixture 5)

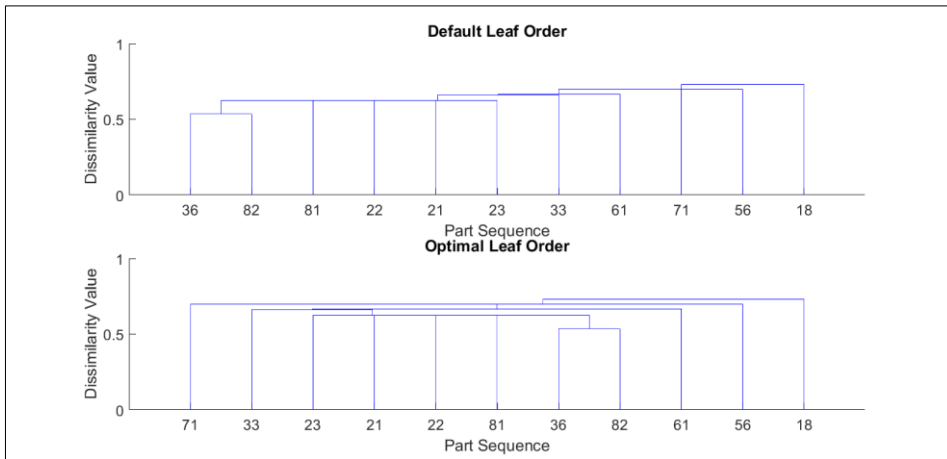


Figure A.40: 100 parts, 10 fixtures (Fixture 6)

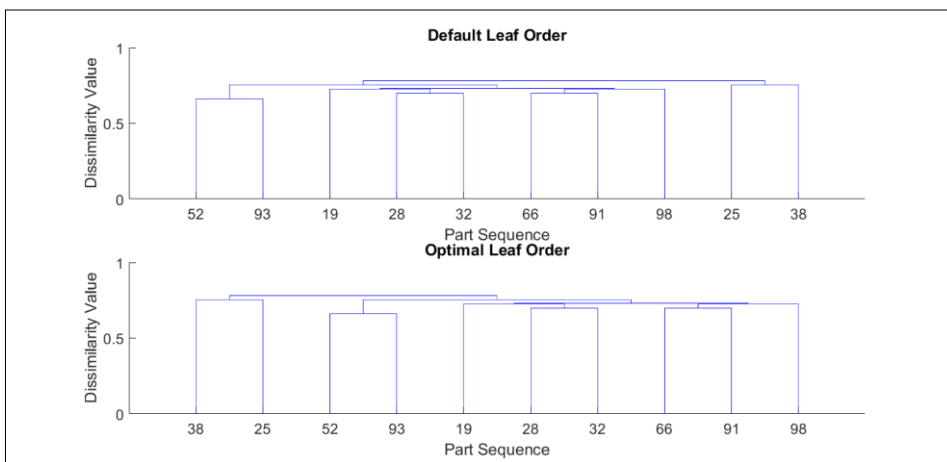


Figure A.41: 100 parts, 10 fixtures (Fixture 7)

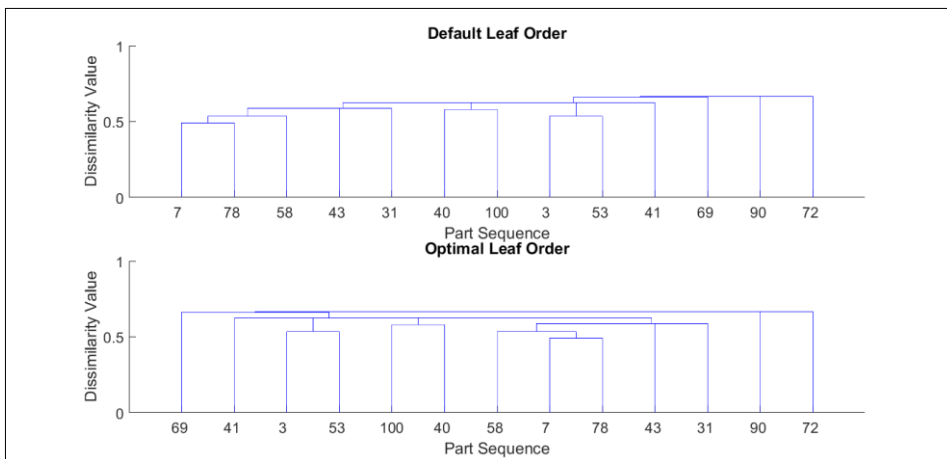


Figure A.42: 100 parts, 10 fixtures (Fixture 8)

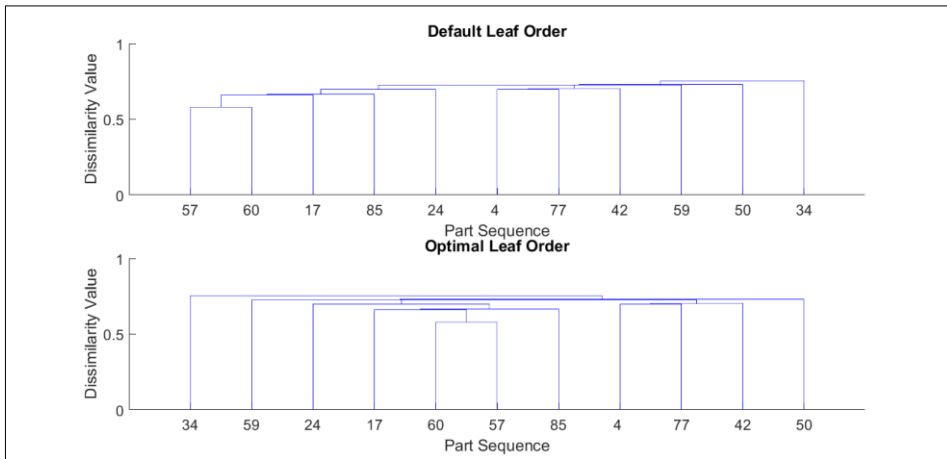


Figure A.43: 100 parts, 10 fixtures (Fixture 9)

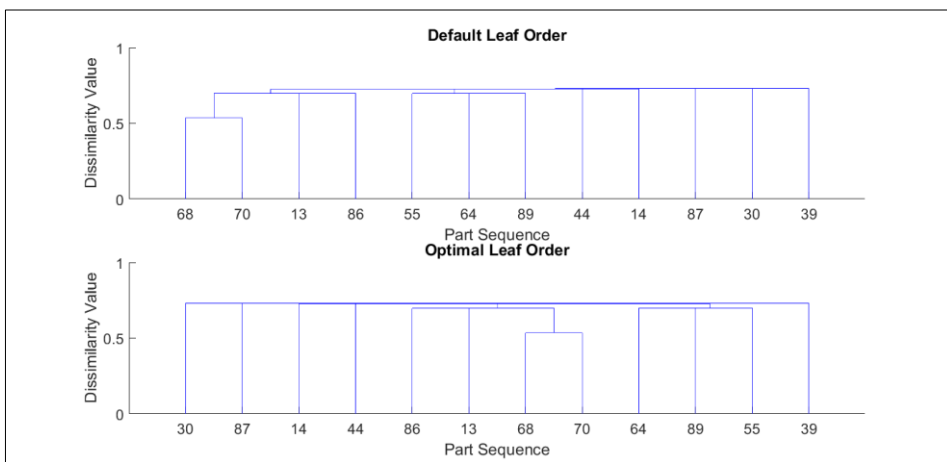


Figure A.44: 100 parts, 10 fixtures (Fixture 10)



Figure A.45: 100 parts, 16 fixtures (Fixture 1)

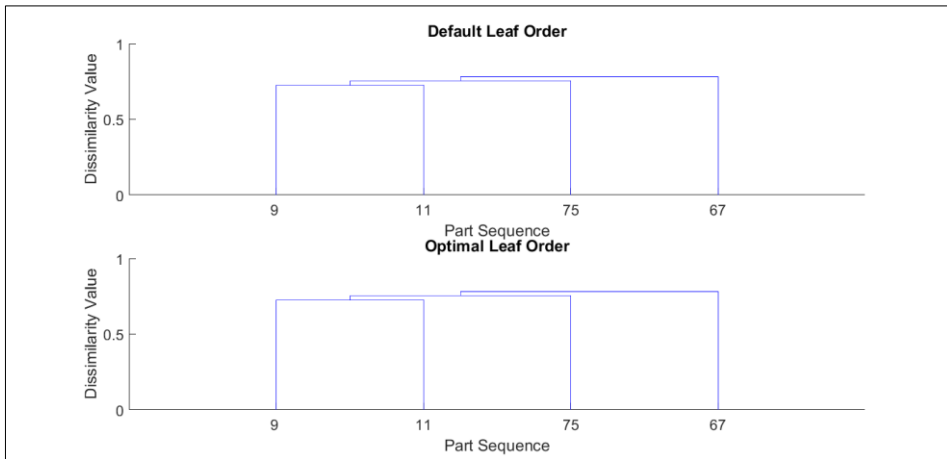


Figure A.46: 100 parts, 16 fixtures (Fixture 2)

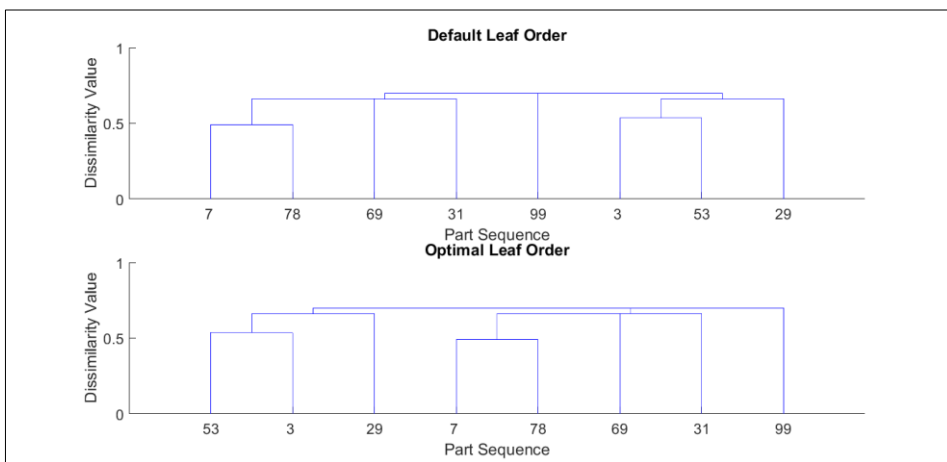


Figure A.47: 100 parts, 16 fixtures (Fixture 3)



Figure A.48: 100 parts, 16 fixtures (Fixture 4)

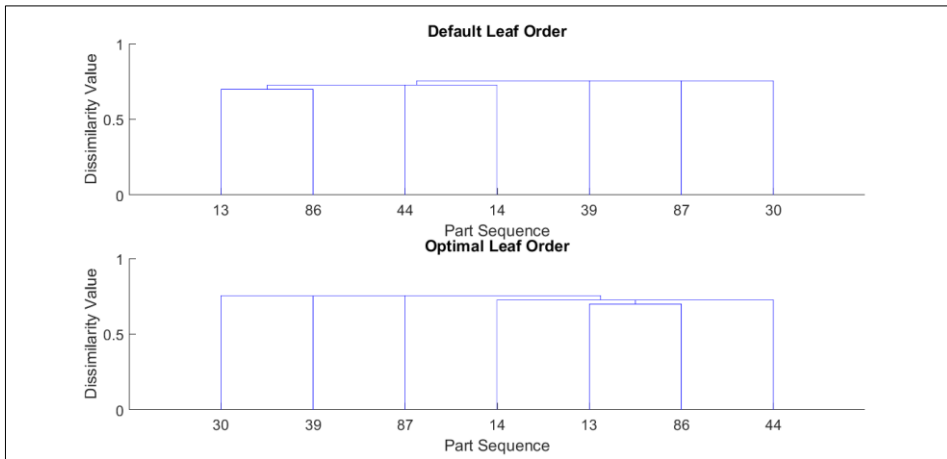


Figure A.49: 100 parts, 16 fixtures (Fixture 5)

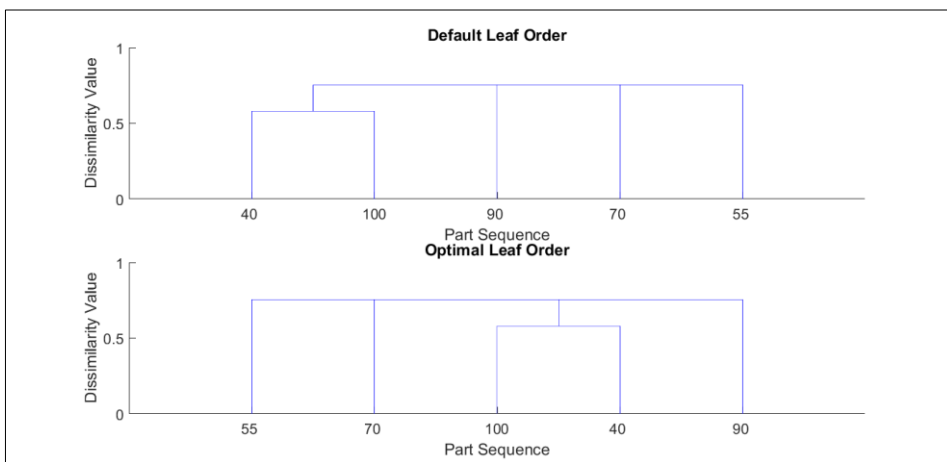


Figure A.50: 100 parts, 16 fixtures (Fixture 6)

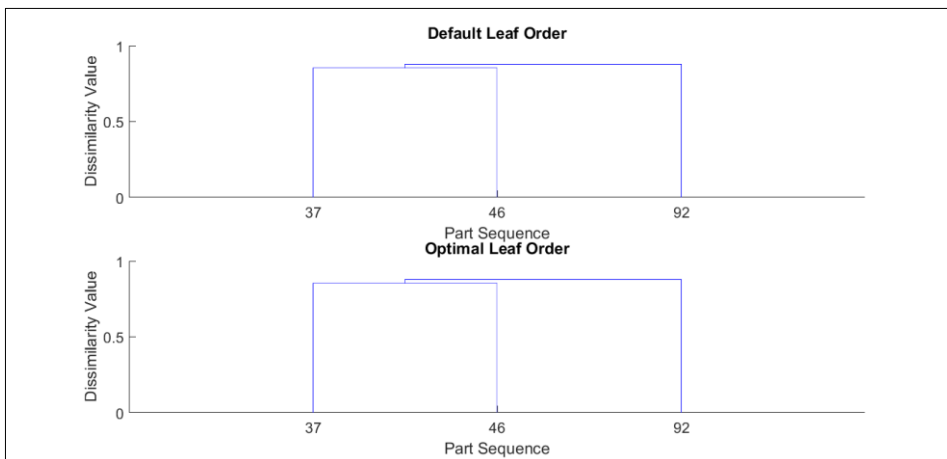


Figure A.51: 100 parts, 16 fixtures (Fixture 7)

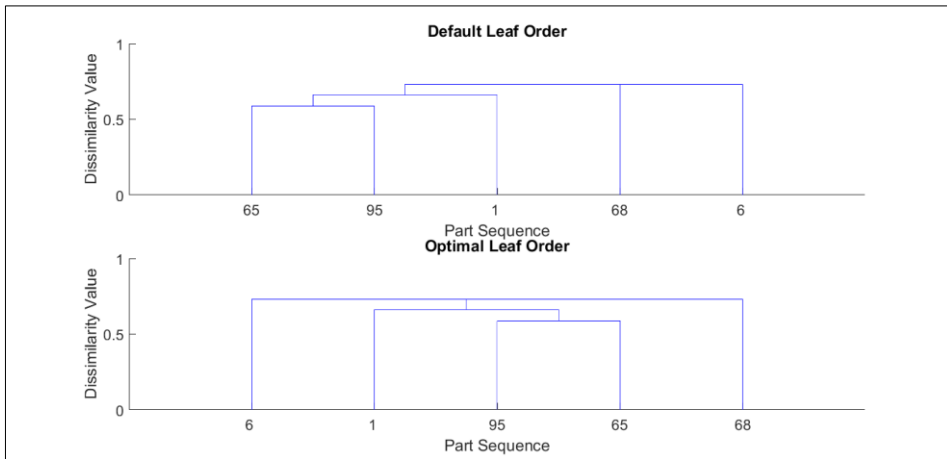


Figure A.52: 100 parts, 16 fixtures (Fixture 8)

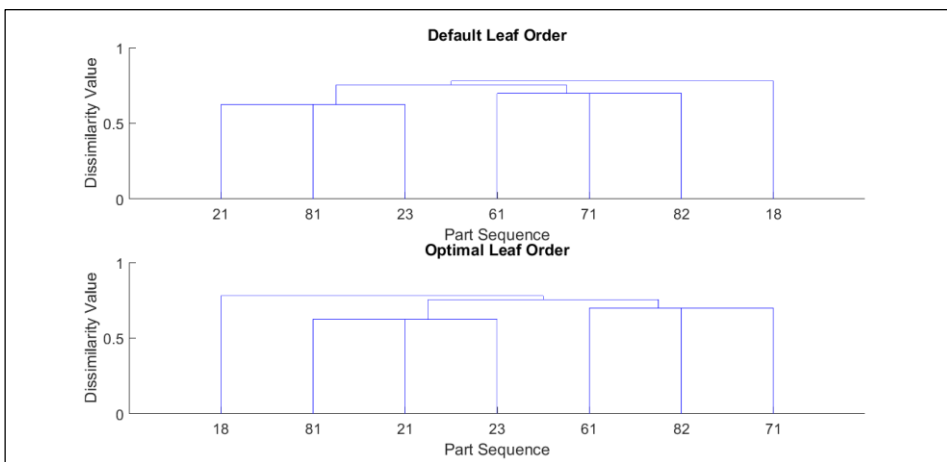


Figure A.53: 100 parts, 16 fixtures (Fixture 9)



Figure A.54: 100 parts, 16 fixtures (Fixture 10)

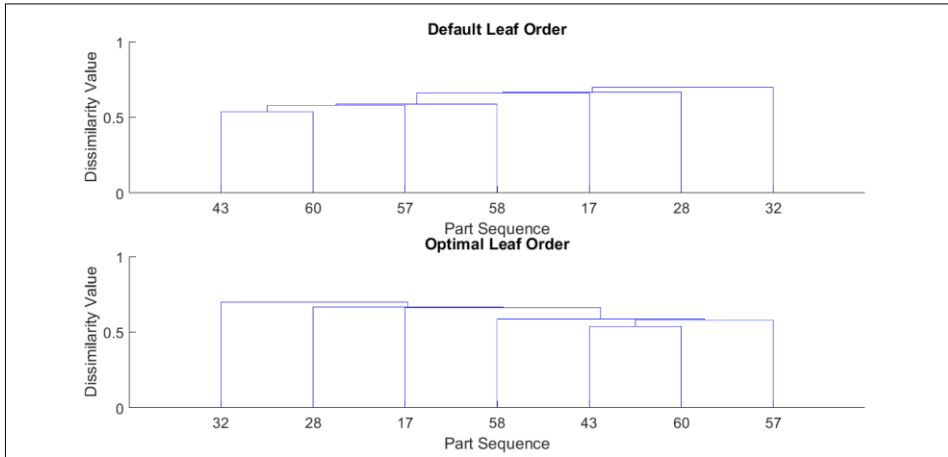


Figure A.55: 100 parts, 16 fixtures (Fixture 11)

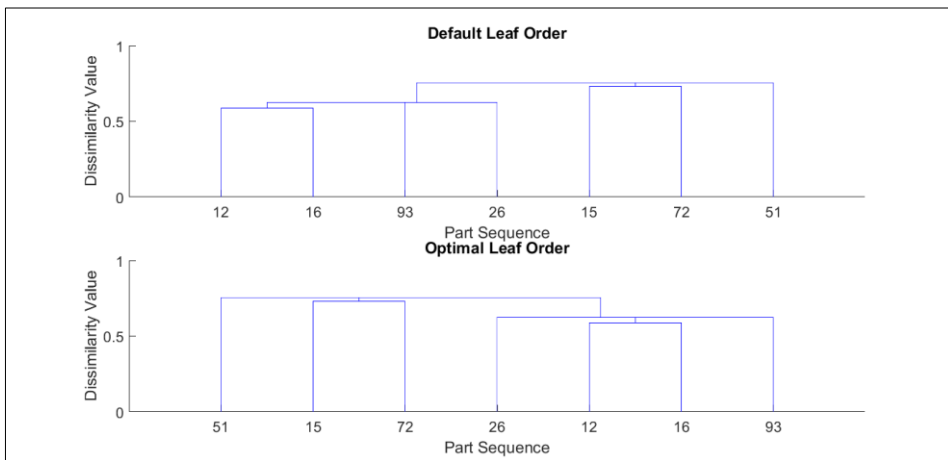


Figure A.56: 100 parts, 16 fixtures (Fixture 12)

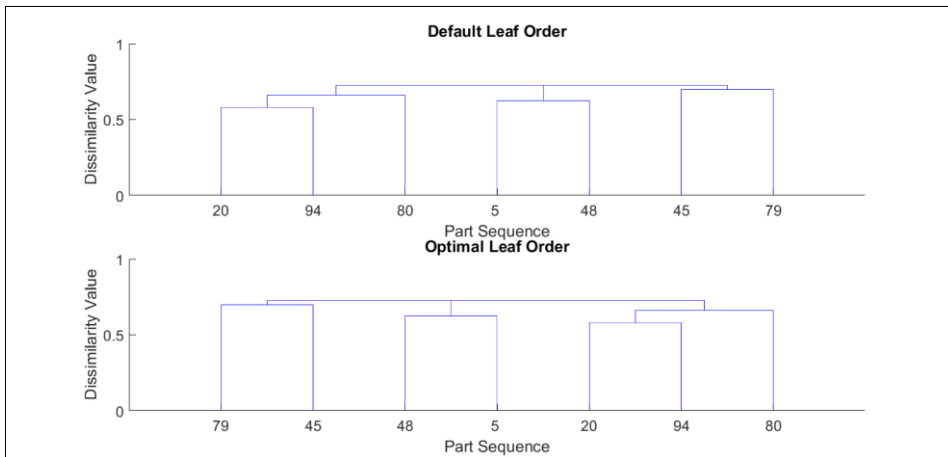


Figure A.57: 100 parts, 16 fixtures (Fixture 13)

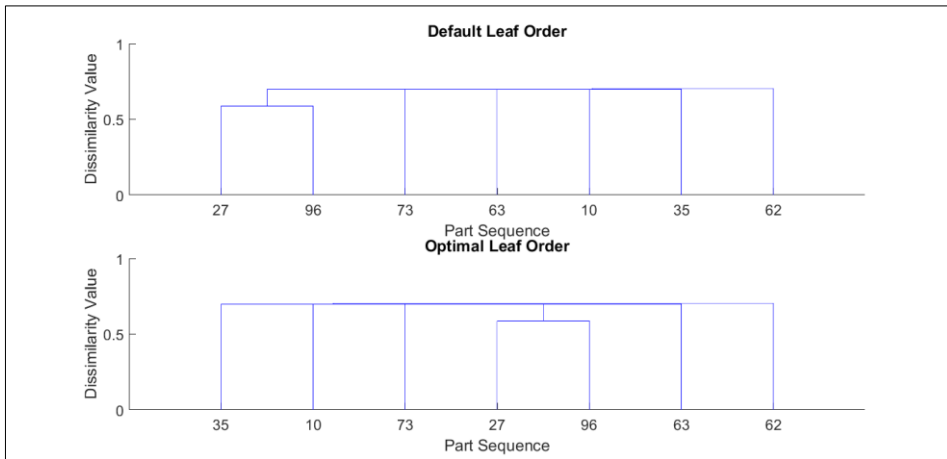


Figure A.58: 100 parts, 16 fixtures (Fixture 14)



Figure A.59: 100 parts, 16 fixtures (Fixture 15)

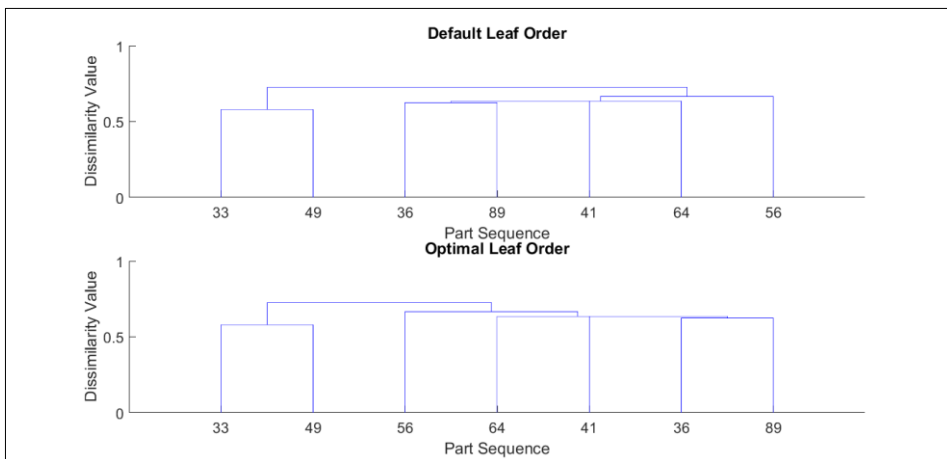


Figure A.60: 100 parts, 16 fixtures (Fixture 16)

A.4. MILP Behaviour

Table A.24 presents the results obtained from the MILP model test results. A total of 25 problem sizes were tested, 3 times each. Convergence to the optimal solution was observed in every case.

Table A.24: MILP test results

Test Problem	Parts	Fixtures	Variables	Relative Gap	Absolute Gap	Feasible Points	Nodes	Constraint Violation	Convergence	Variable Creation Time (s)	Solution Time (s)
1.1	4	2	64	0	0	1	12	0	✓	0.0481	0.8154
1.2	4	2	64	0	0	2	10	0	✓	0.0464	0.9576
1.3	4	2	64	0	0	1	14	0	✓	0.0472	0.8222
2.1	5	2	121	0	0	1	26	0	✓	0.0668	0.8602
2.2	5	2	121	0	0	1	28	0	✓	0.0676	0.8236
2.3	5	2	121	0	0	1	16	0	✓	0.0668	0.6722
3.1	6	2	216	0	0	1	24	0	✓	0.1246	0.8324
3.2	6	2	216	0	0	1	36	0	✓	0.1218	0.8426
3.3	6	2	216	0	0	1	34	0	✓	0.1197	0.8443
4.1	6	3	276	0	0	2	72	0	✓	0.1516	1.057
4.2	6	3	276	0	0	1	24	0	✓	0.1507	0.8248
4.3	6	3	276	0	0	1	56	0	✓	0.1538	0.7546
5.1	7	2	337	0	0	1	64	0	✓	0.2303	0.8926
5.2	7	2	337	0	0	1	68	0	✓	0.2318	0.8767
5.3	7	2	337	0	0	1	120	0	✓	0.2299	0.9392
6.1	7	3	433	0	0	3	214	0	✓	0.305	1.4586
6.2	7	3	433	0	0	1	128	0	✓	0.3076	1.0952
6.3	7	3	433	0	0	1	158	0	✓	0.3114	1.0886
7.1	8	2	512	0	0	1	114	0	✓	0.4662	1.2013
7.2	8	2	512	0	0	1	108	0	✓	0.4684	1.1451
7.3	8	2	512	0	0	1	192	0	✓	0.462	1.3967
8.1	8	3	652	0	0	1	212	1.11E-16	✓	0.6253	1.3766
8.2	8	3	652	0	0	2	308	0	✓	0.636	1.6736
8.3	8	3	652	0	0	2	238	0	✓	0.6248	1.6178
9.1	8	4	736	0	0	3	1416	0	✓	0.8553	4.1837
9.2	8	4	736	0	0	3	302	0	✓	0.7402	1.8341
9.3	8	4	736	0	0	2	194	0	✓	0.7407	1.438
10.1	9	2	721	0	0	1	500	0	✓	0.8693	3.375
10.2	9	2	721	0	0	1	256	0	✓	0.8824	1.9382
10.3	9	2	721	0	0	1	380	0	✓	0.8712	2.5932
11.1	9	3	945	0	0	2	862	0	✓	1.2321	4.7744
11.2	9	3	945	0.0582	0.1667	3	872	0	✓	1.2275	5.321
11.3	9	3	945	0	0	2	684	0	✓	1.2253	4.4668
12.1	9	4	1041	0	0	2	602	0	✓	1.4141	2.9655
12.2	9	4	1041	0.0972	0.1429	2	504	0	✓	1.4171	2.7816
12.3	9	4	1041	0.0822	0.1429	1	1148	0	✓	1.4082	4.9621
13.1	10	2	1000	0.0342	0.1667	1	750	0	✓	1.516	6.9614
13.2	10	2	1000	0	0	2	450	0	✓	1.5028	4.0417
13.3	10	2	1000	0.0431	0.2308	1	550	0	✓	1.4949	4.8609
14.1	10	3	1288	0.0699	0.1111	3	1798	0	✓	2.0591	12.2138
14.2	10	3	1288	0.0693	0.1818	5	3830	0	✓	2.0679	27.5555
14.3	10	3	1288	0.0203	0.0952	2	1600	0	✓	2.0691	14.45
15.1	10	4	1432	0.0866	0.1351	1	1492	0	✓	2.4478	9.6474
15.2	10	4	1432	0.0843	0.1667	3	3552	0	✓	2.4309	21.7063
15.3	10	4	1432	0.0074	0.0667	2	1604	0	✓	2.4361	10.2527
16.1	10	5	1540	0.0997	0.1667	4	3180	0	✓	2.7378	19.3834
16.2	10	5	1540	0	0	2	24852	0	✓	2.7422	94.7919
16.3	10	5	1540	0.0934	0.1807	1	7974	0	✓	2.7086	35.921
17.1	11	2	1321	0	0	1	1492	0	✓	2.5873	16.4751
17.2	11	2	1321	0.0858	0.3333	1	1692	0	✓	2.5669	21.7195
17.3	11	2	1321	0	0	1	1630	0	✓	2.5655	19.7377
18.1	11	3	1721	0.0761	0.1522	1	3290	0	✓	3.635	42.7043
18.2	11	3	1721	0.0729	0.1667	3	3766	0	✓	3.6663	42.4701
18.3	11	3	1721	0.0478	0.1379	3	2974	0	✓	3.6358	32.1482
19.1	11	4	1921	0.0951	0.1667	3	7122	2.22E-16	✓	4.2947	68.0933

19.2	11	4	1921	0.0479	0.0714	3	7586	0	✓	4.3338	50.8467
19.3	11	4	1921	0.0944	0.24	4	18576	0	✓	4.3	153.5086
20.1	11	5	2041	0.0138	0.1515	3	45544	0	✓	4.7242	263.8855
20.2	11	5	2041	0.0832	0.129	4	4160	0	✓	4.7234	32.0126
20.3	11	5	2041	0	0	2	2825	0	✓	4.755	24.8387
21.1	12	2	1728	0.0981	0.5	1	3328	0	✓	4.3486	64.736
21.2	12	2	1728	0	0	1	2299	0	✓	4.3378	45.3214
21.3	12	2	1728	0	0	1	2672	0	✓	4.3209	57.7461
22.1	12	3	2256	0	0	1	15248	0	✓	6.1385	254.3976
22.2	12	3	2256	0	0	2	20144	0	✓	6.1812	353.6451
22.3	12	3	2256	0	0	3	22904	0	✓	6.1816	396.1632
23.1	12	4	2520	0.0919	0.1429	1	12026	0	✓	7.3436	121.3051
23.2	12	4	2520	0.0481	0.15	3	24546	0	✓	7.357	311.4977
23.3	12	4	2520	0.0934	0.1111	2	5684	0	✓	7.3261	104.4132
24.1	12	5	2652	0	0	1	31004	0	✓	7.9055	333.4646
24.2	12	5	2652	0	0	3	330624	0	✓	7.935	3122.298
24.3	12	5	2652	0	0	4	60554	0	✓	7.9548	621.5673
25.1	12	6	2784	0	0	2	17560	1.78E-15	✓	8.6281	185.0378
25.2	12	6	2784	0.0958	0.0851	3	6682	0	✓	9.1238	92.8686
25.3	12	6	2784	0	0	3	28872	0	✓	8.5869	324.2812

A.4.1. MILP Convergence Graphs

Figure A.61 to Figure A.135 display the B&B convergence graphs for the MILP problems solved in Table A.24.

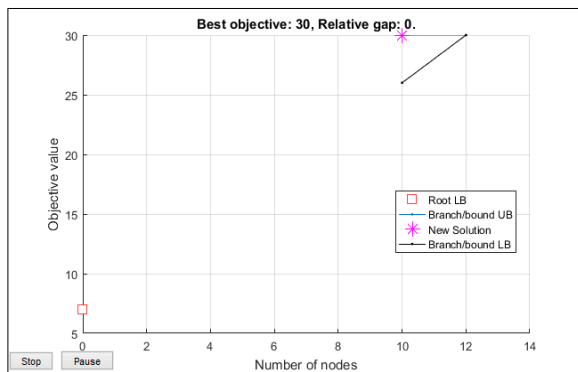


Figure A.61: 4 parts, 2 fixtures (1)

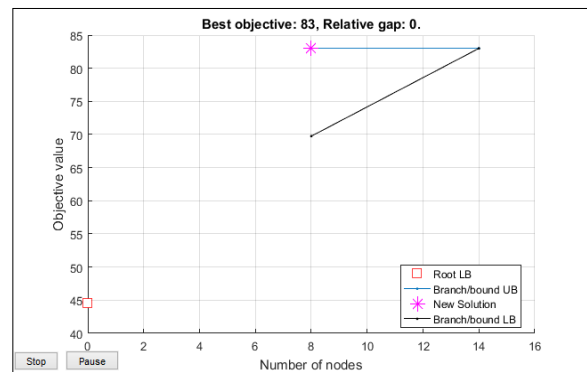


Figure A.63: 4 parts, 2 fixtures (3)

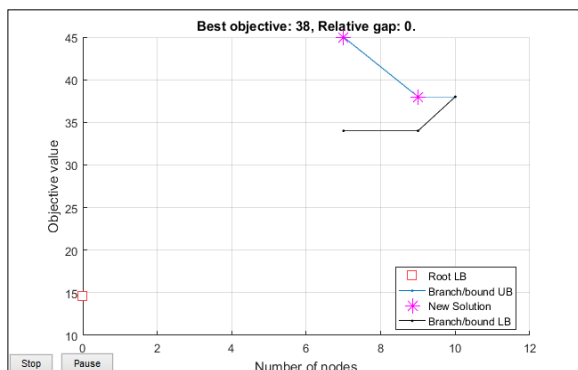


Figure A.62: 4 parts, 2 fixtures (2)

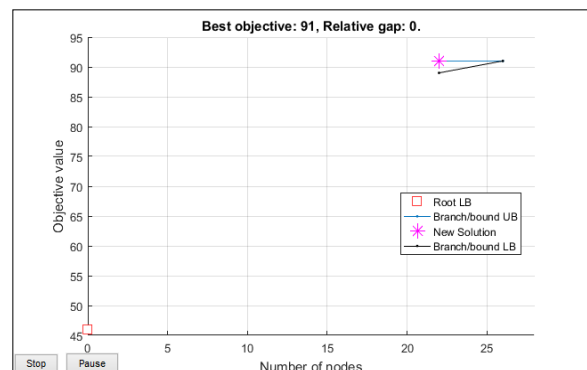


Figure A.64: 5 parts, 2 fixtures (1)

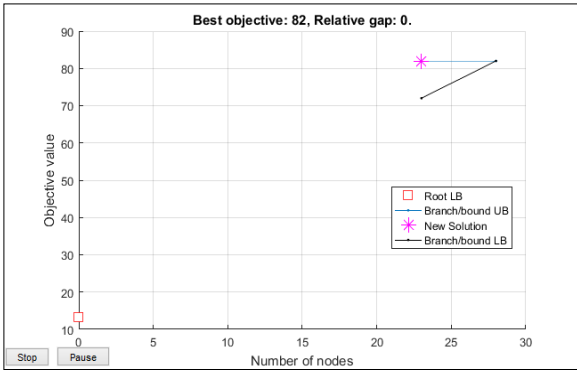


Figure A.65: 5 parts, 2 fixtures (2)

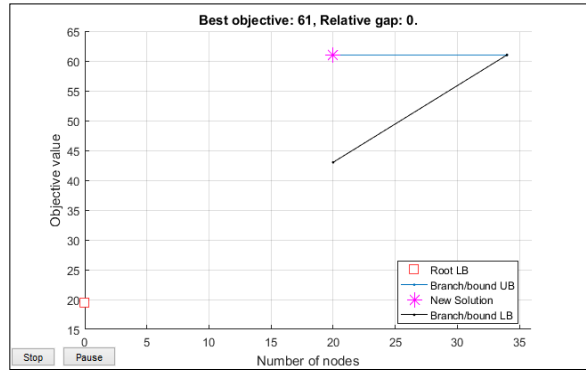


Figure A.69: 6 parts, 2 fixtures (3)

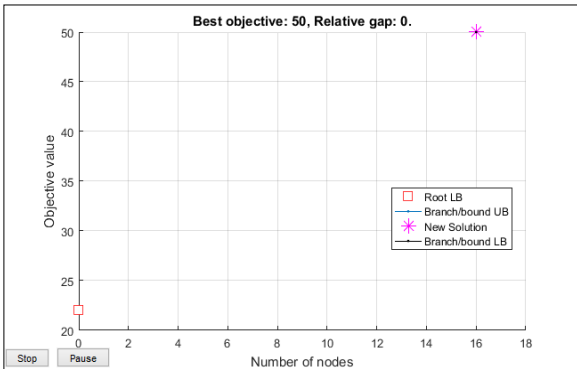


Figure A.66: 5 parts, 2 fixtures (3)

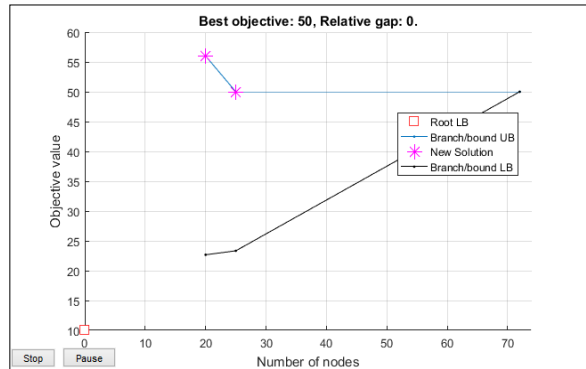


Figure A.70: 6 parts, 3 fixtures (1)

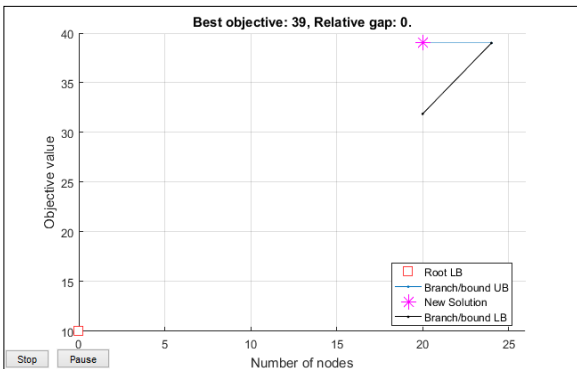


Figure A.67: 6 parts, 2 fixtures (1)

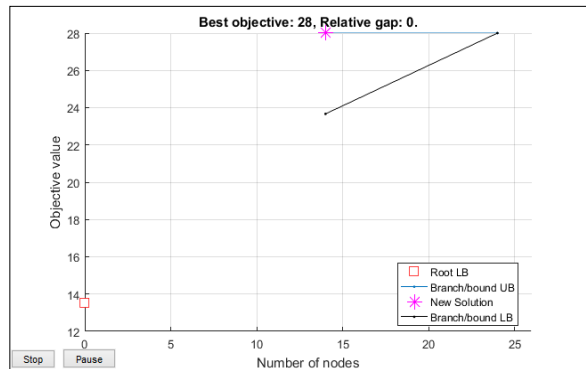


Figure A.71: 6 parts, 3 fixtures (2)

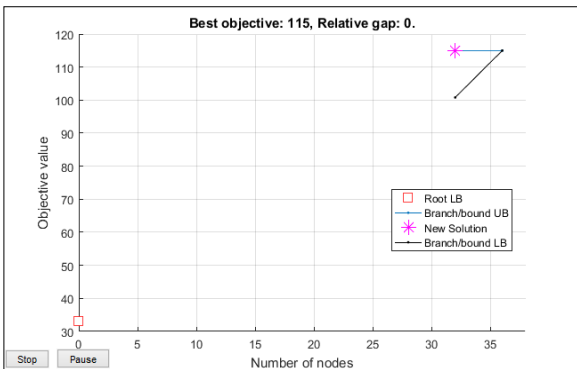


Figure A.68: 6 parts, 2 fixtures (2)

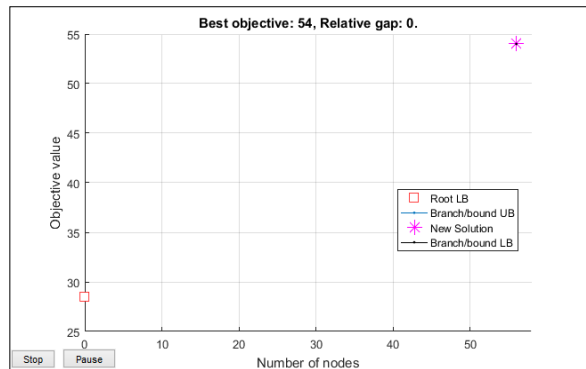


Figure A.72: 6 parts, 3 fixtures (3)

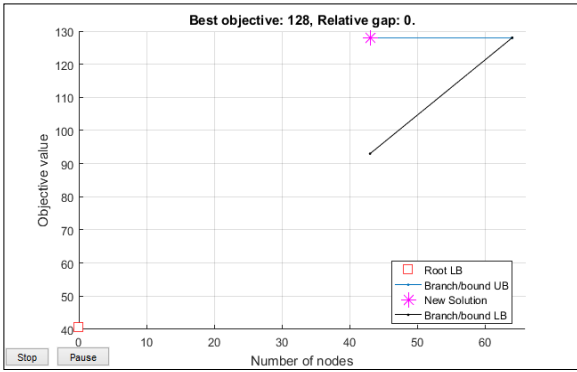


Figure A.73: 7 parts, 2 fixtures (1)

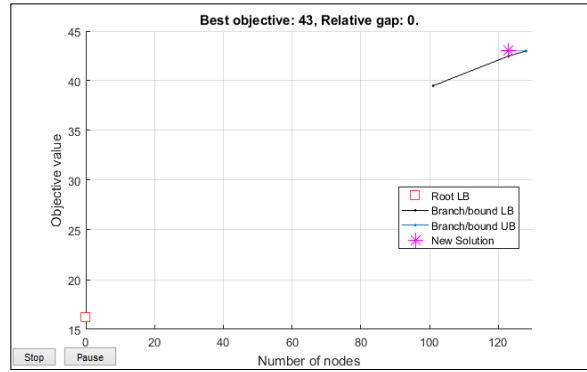


Figure A.77: 7 parts, 3 fixtures (2)

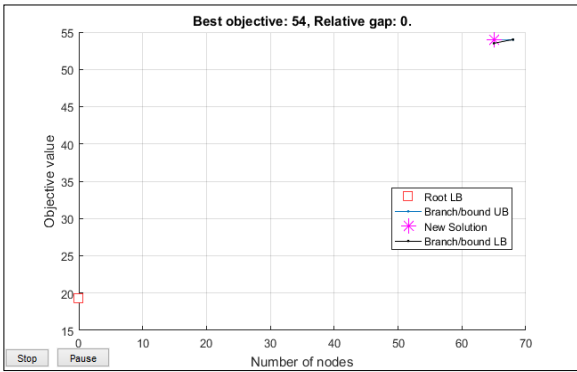


Figure A.74: 7 parts, 2 fixtures (2)

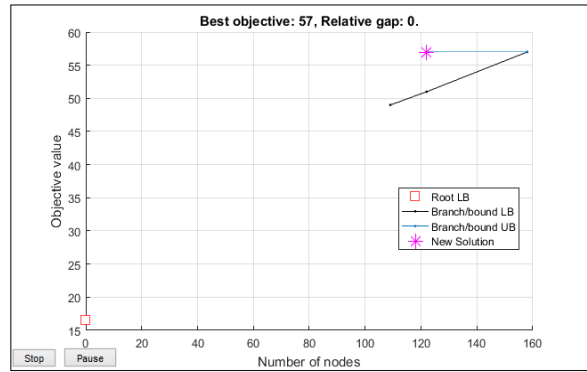


Figure A.78: 7 parts, 3 fixtures (3)

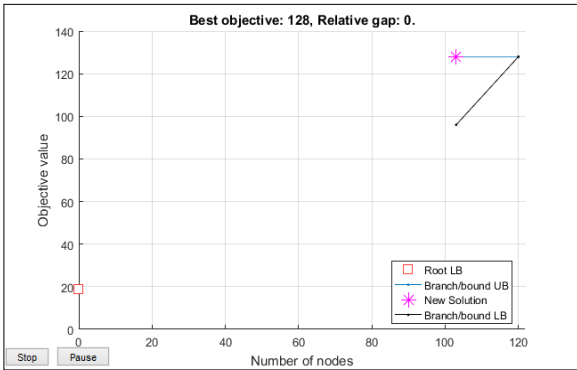


Figure A.75: 7 parts, 2 fixtures (3)

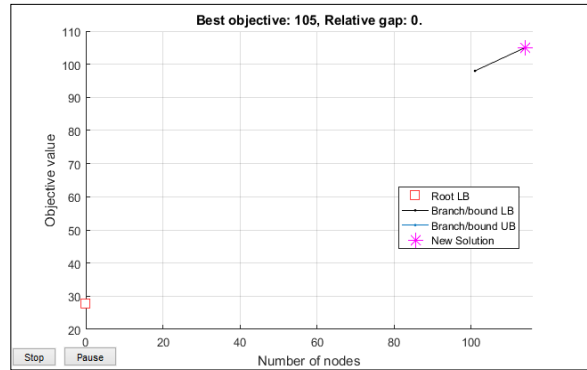


Figure A.79: 8 parts, 2 fixtures (1)

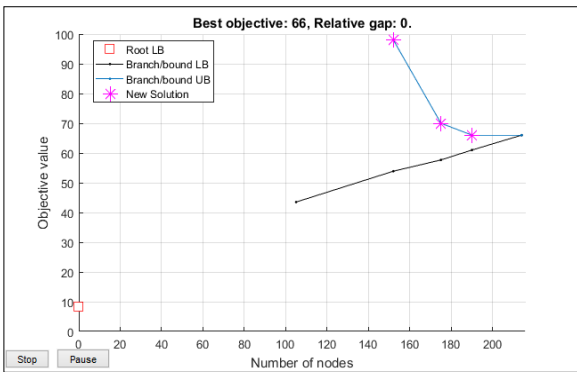


Figure A.76: 7 parts, 3 fixtures (1)

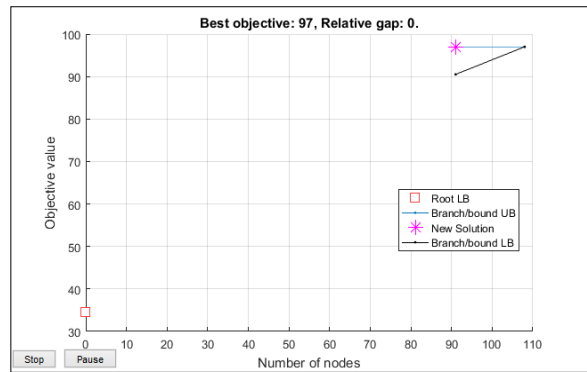


Figure A.80: 8 parts, 2 fixtures (2)

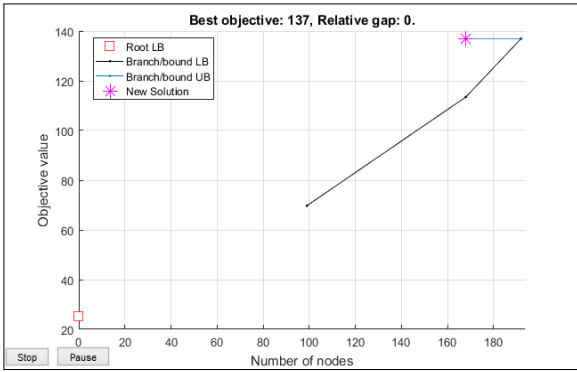


Figure A.81: 8 parts, 2 fixtures (3)

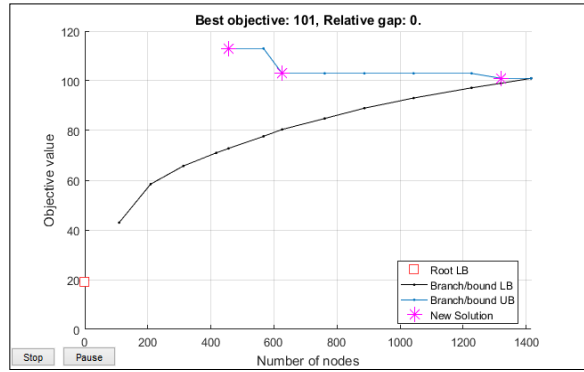


Figure A.85: 8 parts, 4 fixtures (1)

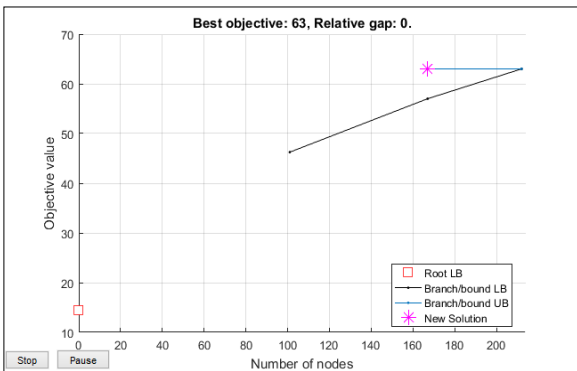


Figure A.82: 8 parts, 3 fixtures (1)

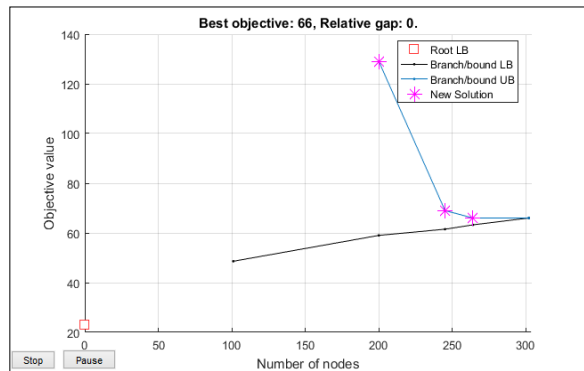


Figure A.86: 8 parts, 4 fixtures (2)

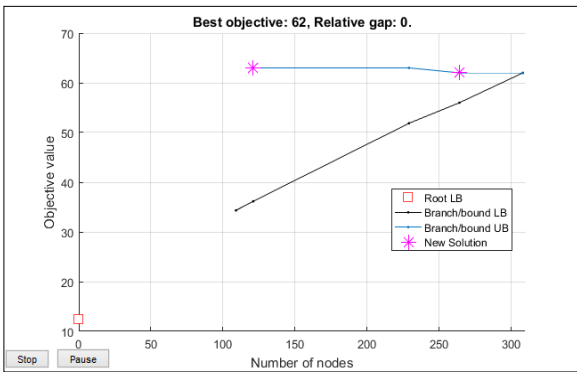


Figure A.83: 8 parts, 3 fixtures (2)

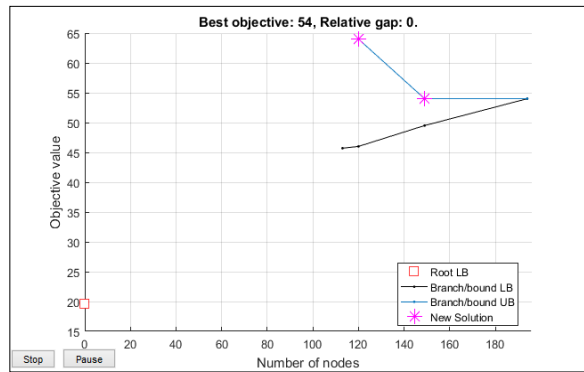


Figure A.87: 8 parts, 4 fixtures (3)

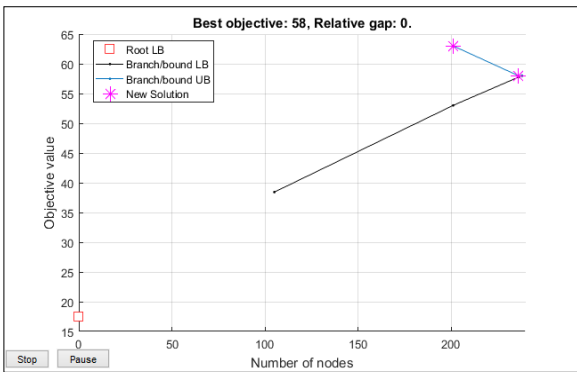


Figure A.84: 8 parts, 3 fixtures (3)

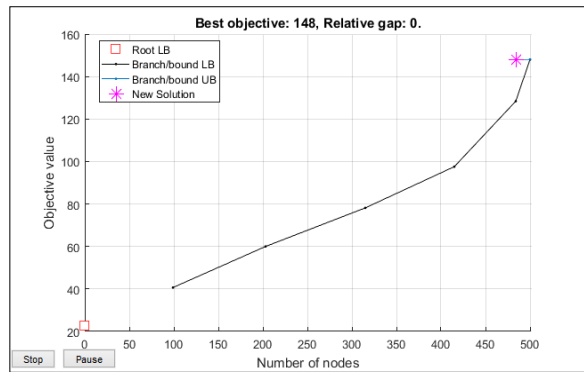


Figure A.88: 9 parts, 2 fixtures (1)

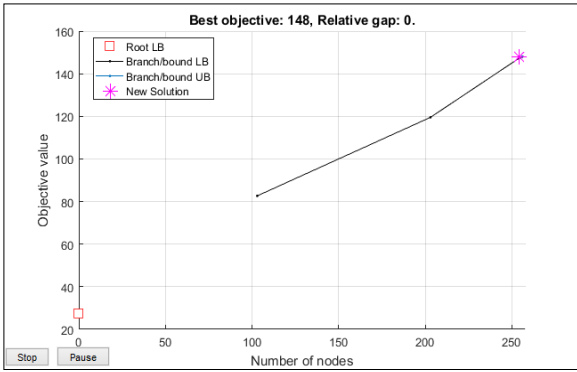


Figure A.89: 9 parts, 2 fixtures (2)

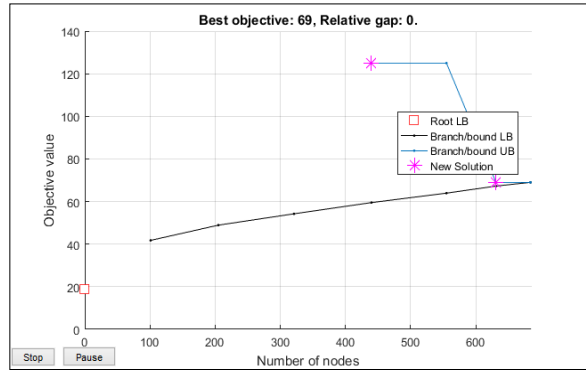


Figure A.93: 9 parts, 3 fixtures (3)

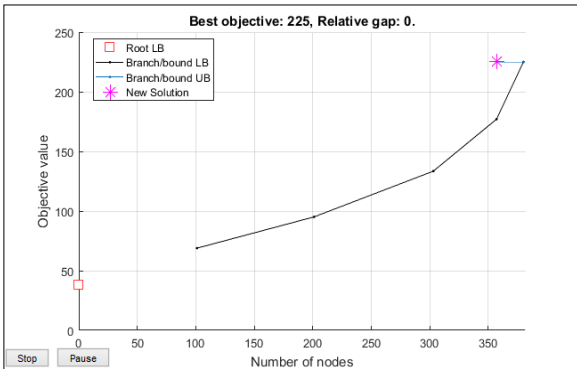


Figure A.90: 9 parts, 2 fixtures (3)

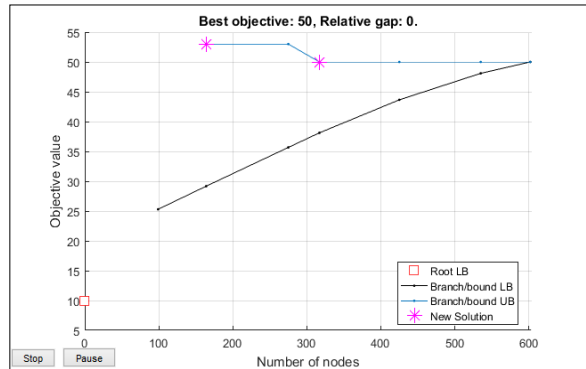


Figure A.94: 9 parts, 4 fixtures (1)

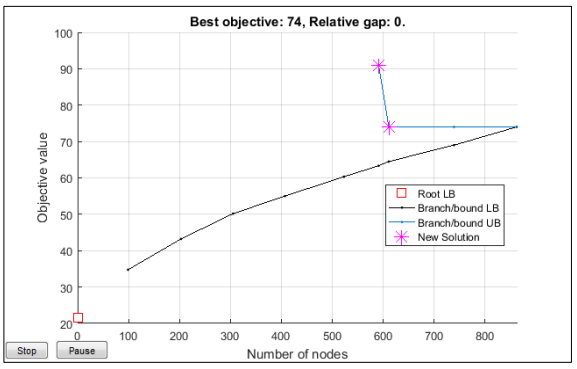


Figure A.91: 9 parts, 3 fixtures (1)

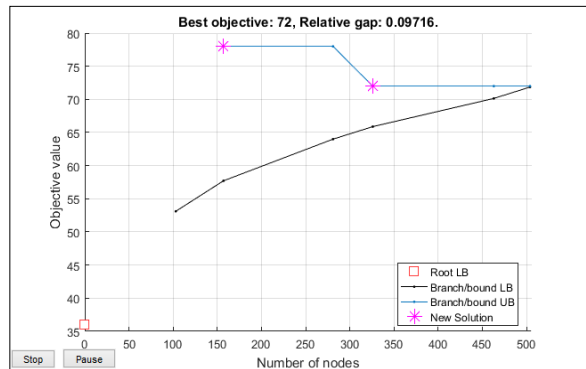


Figure A.95: 9 parts, 4 fixtures (2)

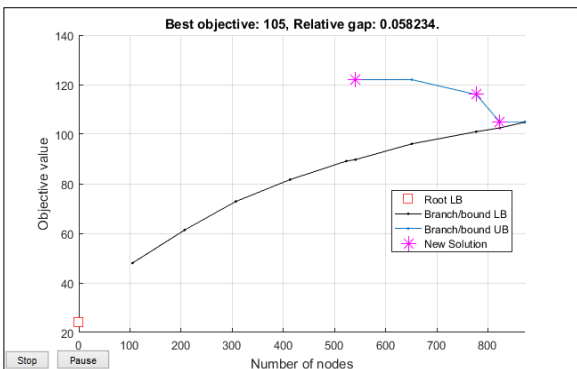


Figure A.92: 9 parts, 3 fixtures (2)

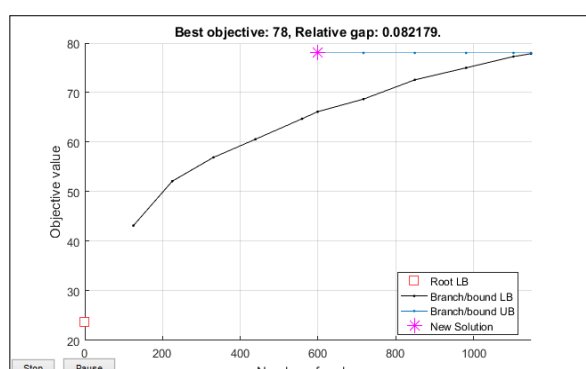


Figure A.96: 9 parts, 4 fixtures (3)

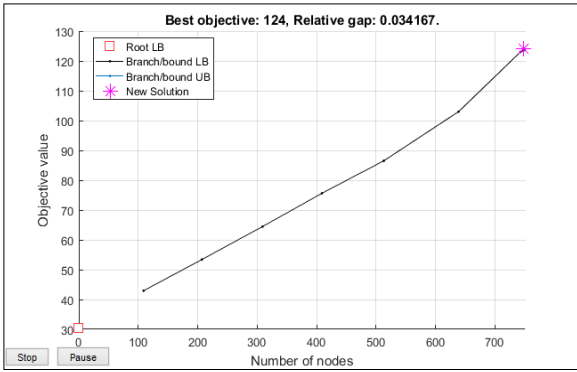


Figure A.97: 10 parts, 2 fixtures (1)

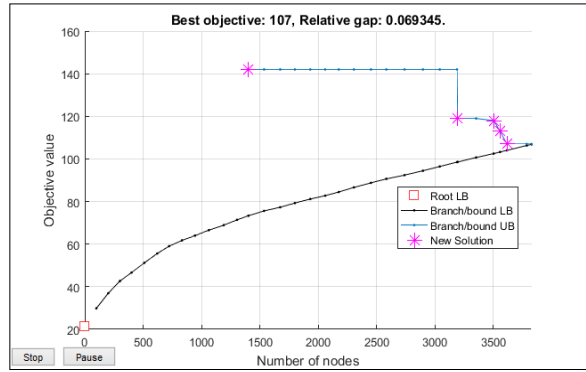


Figure A.101: 10 parts, 3 fixtures (2)

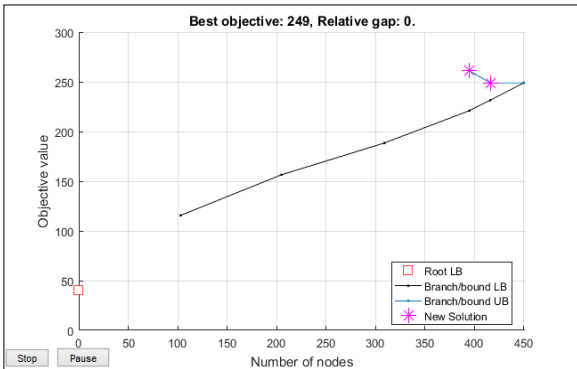


Figure A.98: 10 parts, 2 fixtures (2)

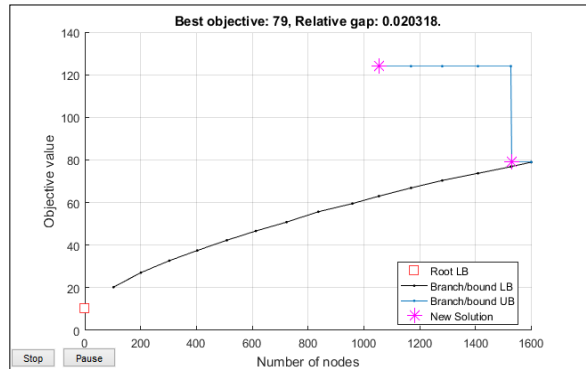


Figure A.102: 10 parts, 3 fixtures (3)

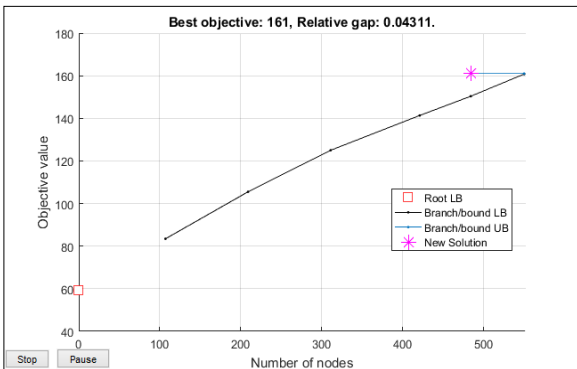


Figure A.99: 10 parts, 2 fixtures (3)

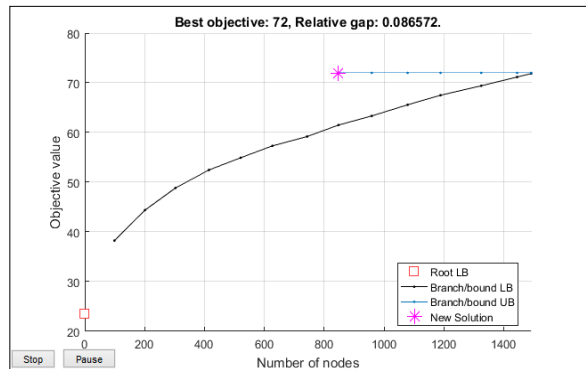


Figure A.103: 10 parts, 4 fixtures (1)

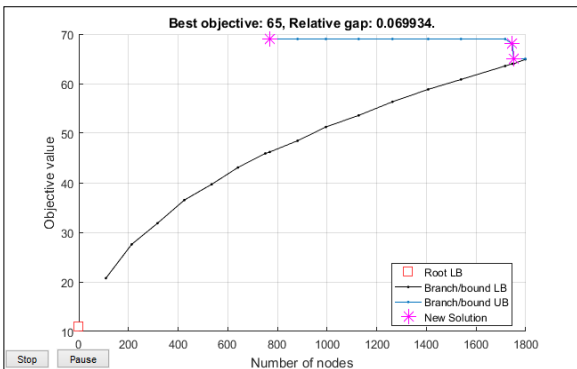


Figure A.100: 10 parts, 3 fixtures (1)

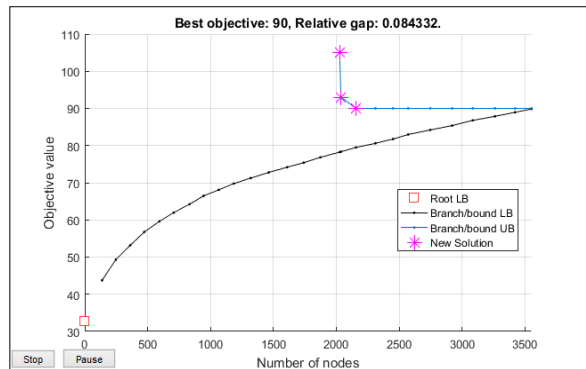


Figure A.104: 10 parts, 4 fixtures (2)

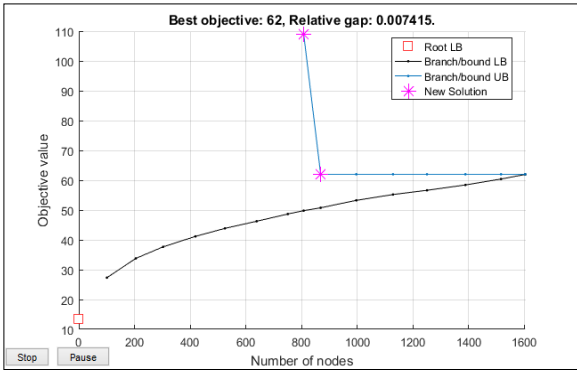


Figure A.105: 10 parts, 4 fixtures (3)

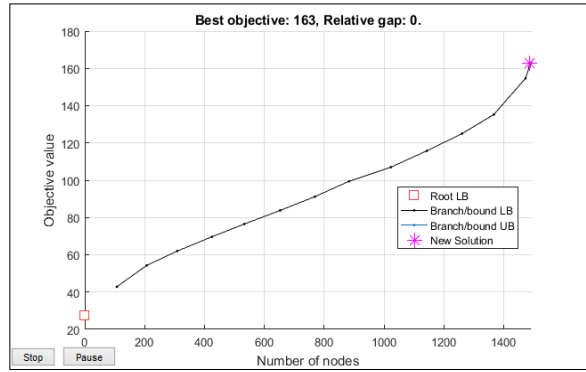


Figure A.109: 11 parts, 2 fixtures (1)

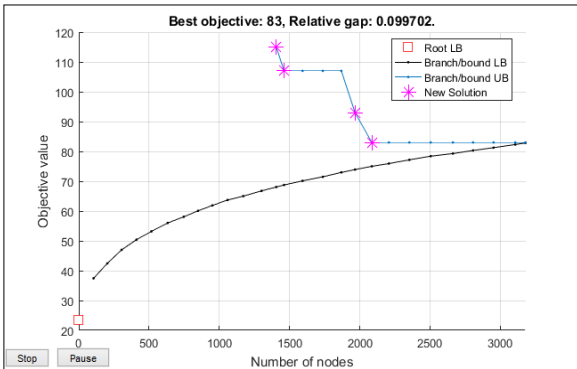


Figure A.106: 10 parts, 5 fixtures (1)

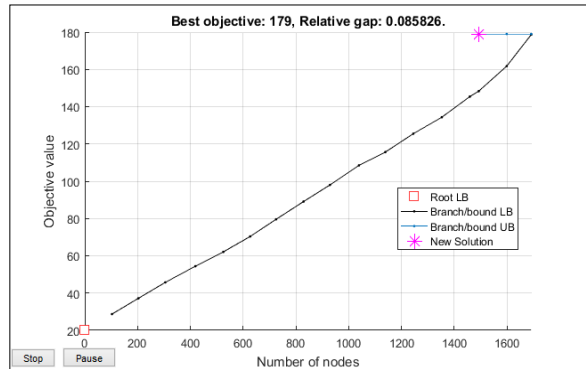


Figure A.110: 11 parts, 2 fixtures (2)

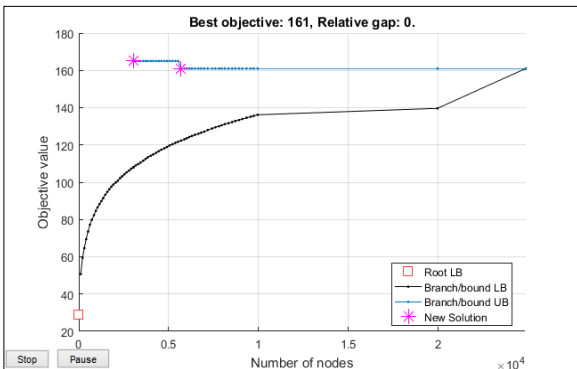


Figure A.107: 10 parts, 5 fixtures (2)

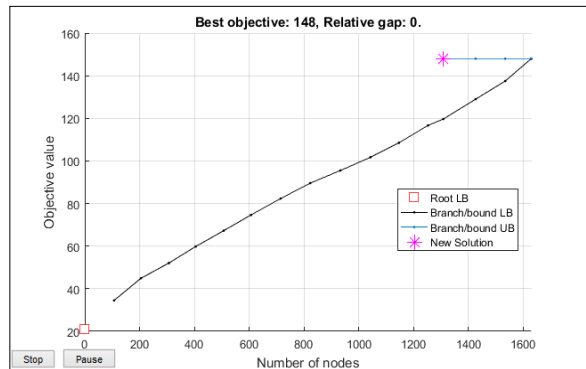


Figure A.111: 11 parts, 2 fixtures (3)

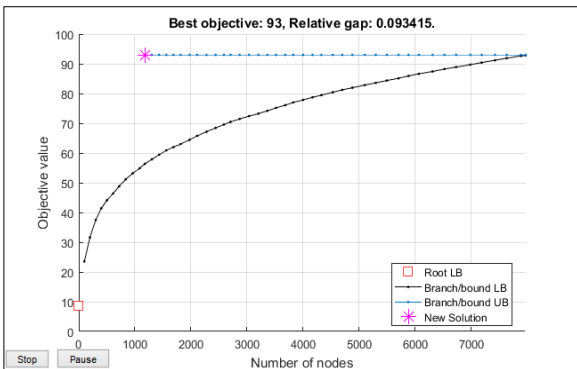


Figure A.108: 10 parts, 5 fixtures (3)

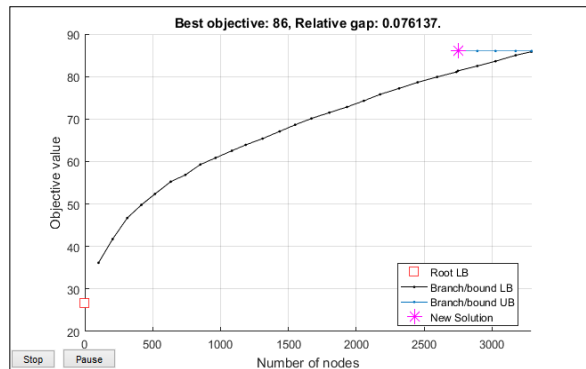


Figure A.112: 11 parts, 3 fixtures (1)

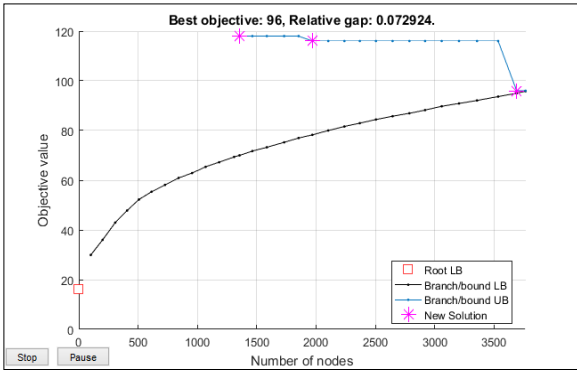


Figure A.113: 11 parts, 3 fixtures (2)

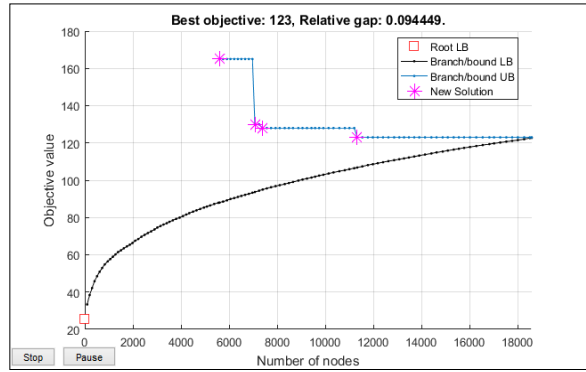


Figure A.117: 11 parts, 4 fixtures (3)

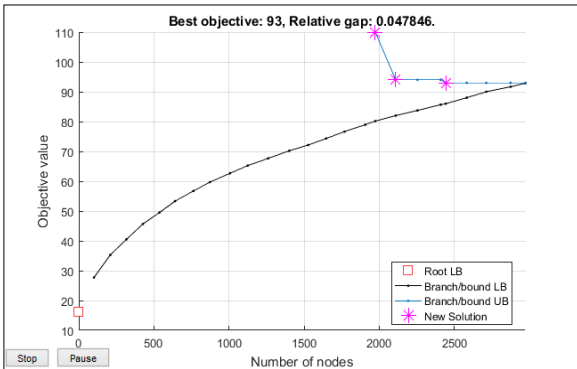


Figure A.114: 11 parts, 3 fixtures (3)

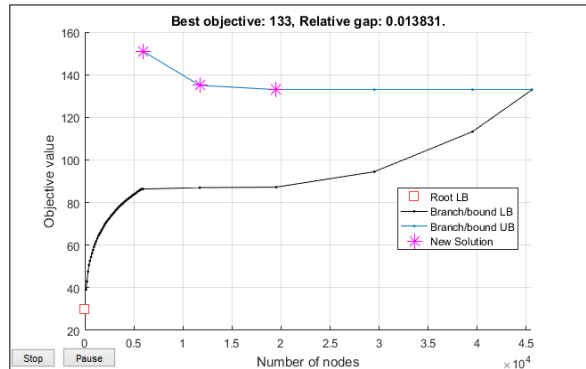


Figure A.118: 11 parts, 5 fixtures (1)

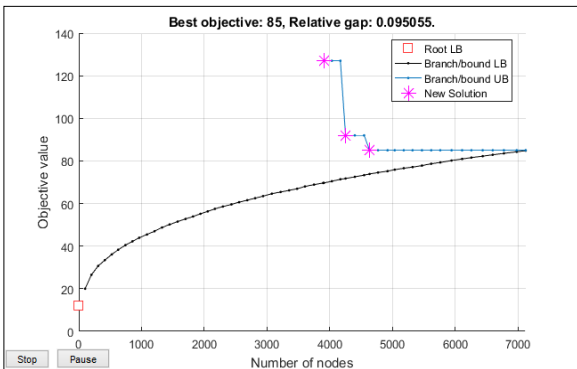


Figure A.115: 11 parts, 4 fixtures (1)

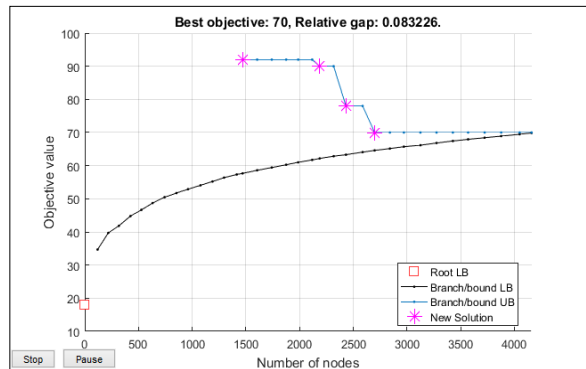


Figure A.119: 11 parts, 5 fixtures (2)

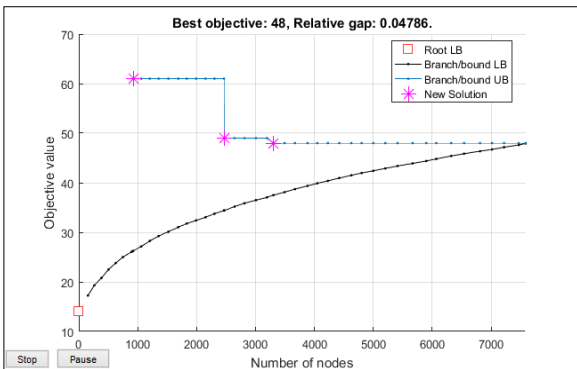


Figure A.116: 11 parts, 4 fixtures (2)

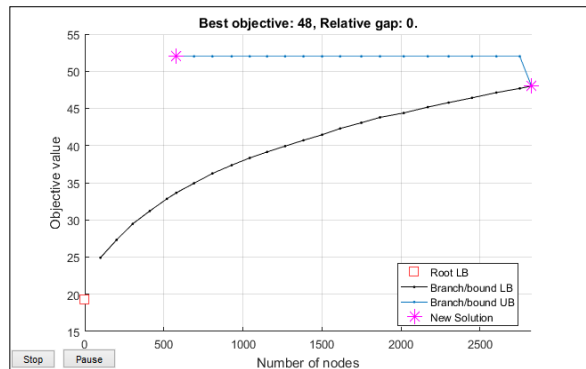


Figure A.120: 11 parts, 5 fixtures (3)

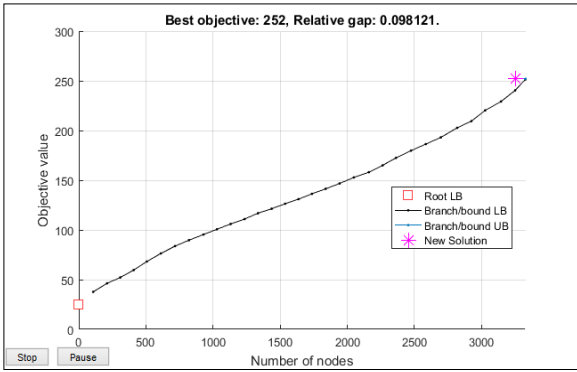


Figure A.121: 12 parts, 2 fixtures (1)

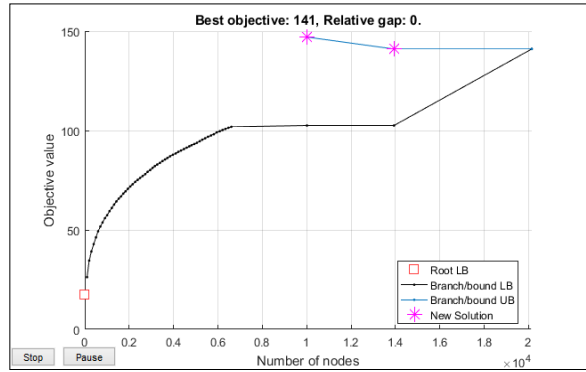


Figure A.125: 12 parts, 3 fixtures (2)

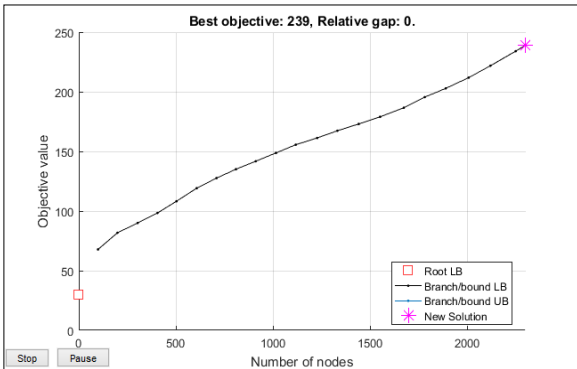


Figure A.122: 12 parts, 2 fixtures (2)

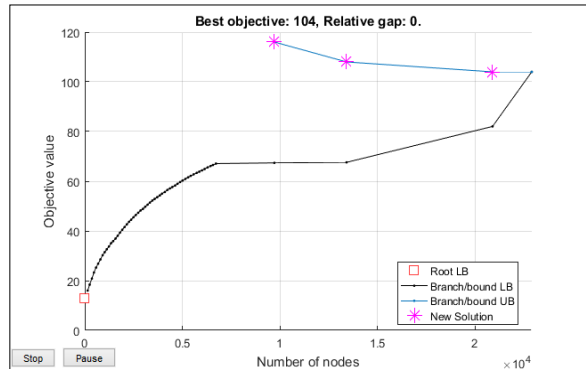


Figure A.126: 12 parts, 3 fixtures (3)

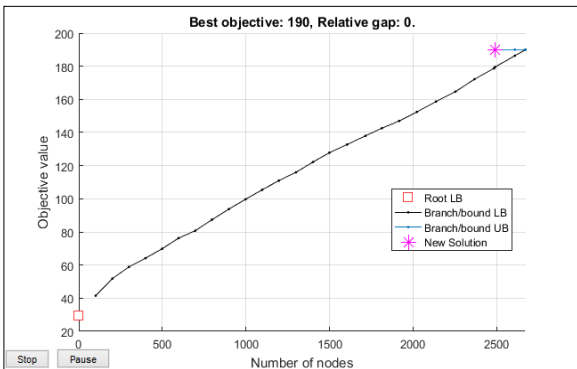


Figure A.123: 12 parts, 2 fixtures (3)

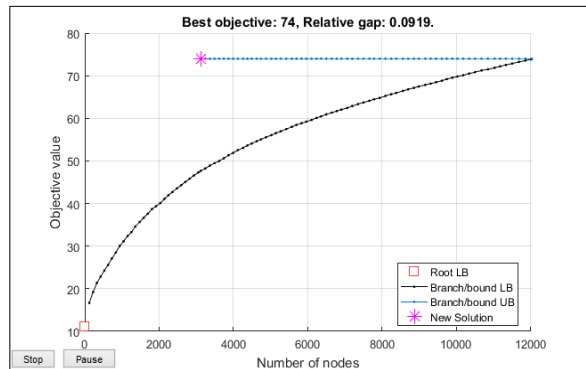


Figure A.127: 12 parts, 4 fixtures (1)

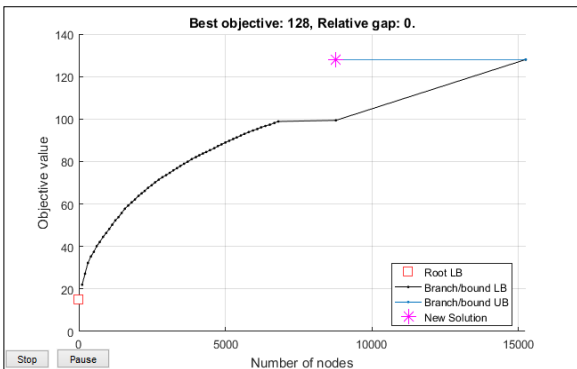


Figure A.124: 12 parts, 3 fixtures (1)

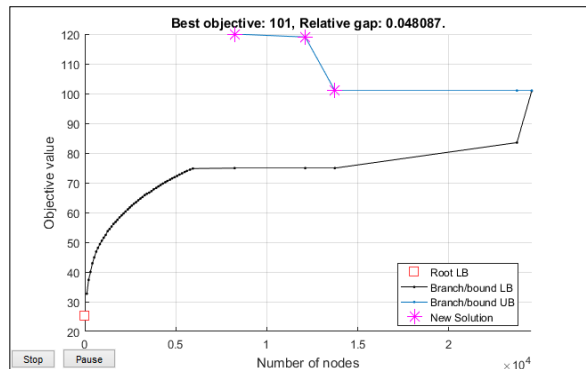


Figure A.128: 12 parts, 4 fixtures (2)

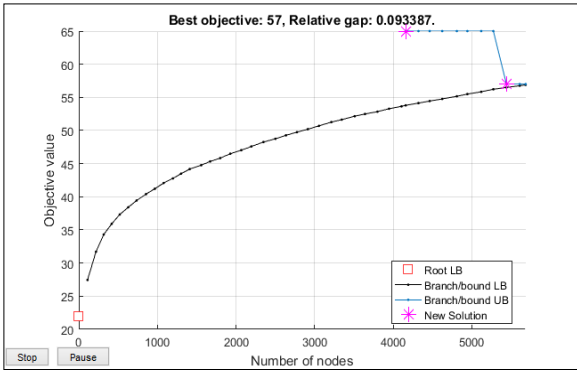


Figure A.129: 12 parts, 4 fixtures (3)

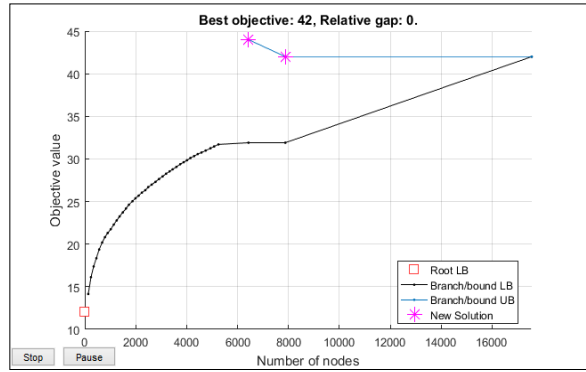


Figure A.133: 12 parts, 6 fixtures (1)

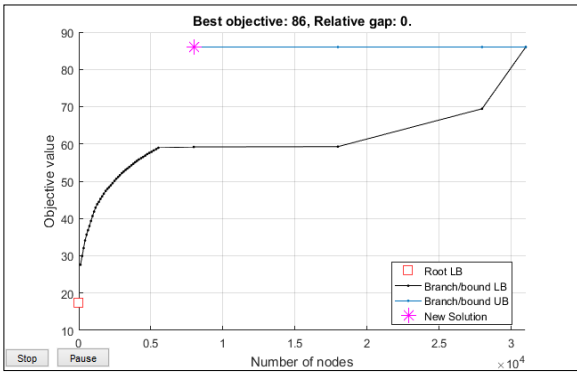


Figure A.130: 12 parts, 5 fixtures (1)

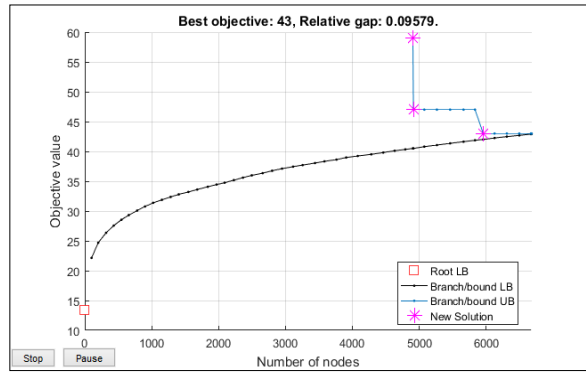


Figure A.134: 12 parts, 6 fixtures (2)

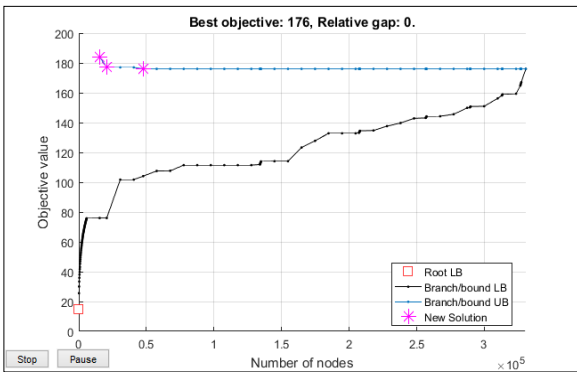


Figure A.131: 12 parts, 5 fixtures (2)

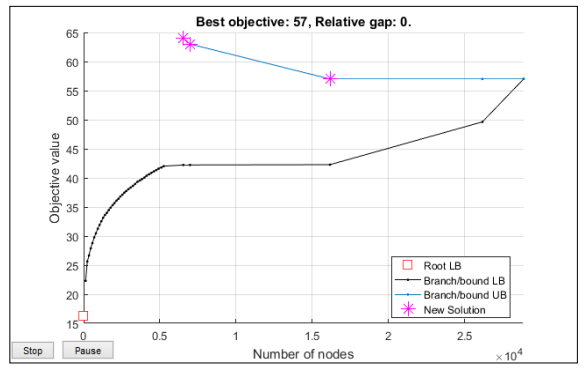


Figure A.135: 12 parts, 6 fixtures (3)

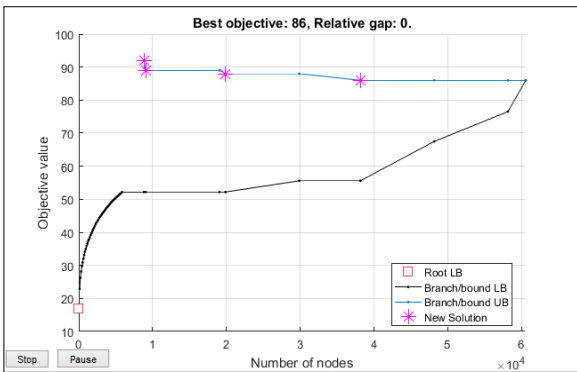


Figure A.132: 12 parts, 5 fixtures (3)

A.5. Heuristic Solution Quality

Table A.25 presents the S3H results in comparison to the MILP results for the same problems. The test problems used are analogous to those in Table A.24.

Table A.25: Stage III Heuristic/MILP model comparison test results

Test Problem	Parts	Fixtures	Variables	S3H solution	MILP solution	Solution Increase (%)	S3H solution time	MILP solution time
1.1	4	2	64	31	30	3.333	0.0103	0.8154
1.2	4	2	64	38	38	0	0.0081	0.9576
1.3	4	2	64	83	83	0	0.0073	0.8222
3.1	6	2	216	90	39	130.77	0.0119	0.8324
3.2	6	2	216	115	115	0	0.0153	0.8426
3.3	6	2	216	61	61	0	0.0116	0.8443
4.1	6	3	276	86	50	72	0.0127	1.057
4.2	6	3	276	29	28	3.57	0.0139	0.8248
4.3	6	3	276	93	54	72.22	0.0146	0.7546
7.1	8	2	512	177	105	68.57	0.0124	1.2013
7.2	8	2	512	97	97	0	0.0156	1.1451
7.3	8	2	512	140	137	2.19	0.0124	1.3967
9.1	8	4	736	148	101	46.53	0.012	4.1837
9.2	8	4	736	69	66	4.55	0.0122	1.8341
9.3	8	4	736	114	54	111.11	0.012	1.438
11.1	9	3	945	81	74	9.46	0.0145	4.7744
11.2	9	3	945	147	105	40	0.0159	5.321
11.3	9	3	945	73	69	5.80	0.014	4.4668
13.1	10	2	1000	170	124	37.10	0.0163	6.9614
13.2	10	2	1000	261	249	4.82	0.0156	4.0417
13.3	10	2	1000	161	161	0	0.0162	4.8609
16.1	10	5	1540	154	83	85.54	0.015	19.3834
16.2	10	5	1540	186	161	15.53	0.0119	94.7919
16.3	10	5	1540	156	93	67.74	0.0121	35.921
21.1	12	2	1728	252	252	0	0.0177	64.736
21.2	12	2	1728	239	239	0	0.0122	45.3214
21.3	12	2	1728	190	190	0	0.0123	57.7461
22.1	12	3	2256	151	128	17.97	0.014	254.3976
22.2	12	3	2256	187	141	32.62	0.0153	353.6451
22.3	12	3	2256	143	104	37.5	0.0151	396.1632
23.1	12	4	2520	141	74	90.54	0.0156	121.3051
23.2	12	4	2520	129	101	27.72	0.0074	311.4977
23.3	12	4	2520	114	57	100	0.0124	104.4132
25.1	12	6	2784	105	42	150	0.0153	185.0378
25.2	12	6	2784	83	43	93.02	0.0142	92.8686
25.3	12	6	2784	123	57	115.79	0.0141	324.2812

A.6. Simulation Behaviour

This section presents the idle time and utilisation graphs from the simulation GUI.

A.6.1. Idle Time Graphs

Figure A.136 to Figure A.151 present the idle time graphs for the problems tested; Idle Time (s) vs. Simulation Runtime (s) for every test. The graphs were separated for cellFR and cellPP for each test.

100 parts, 5 fixtures:

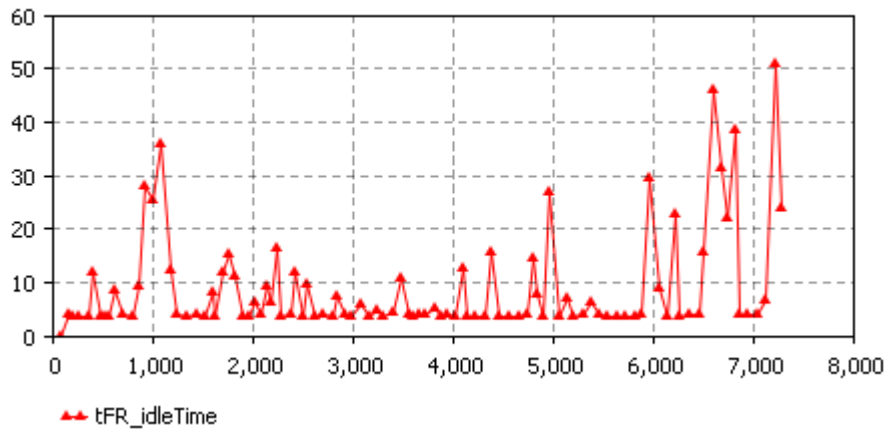


Figure A.136: cellFR (100 parts, 5 fixtures)

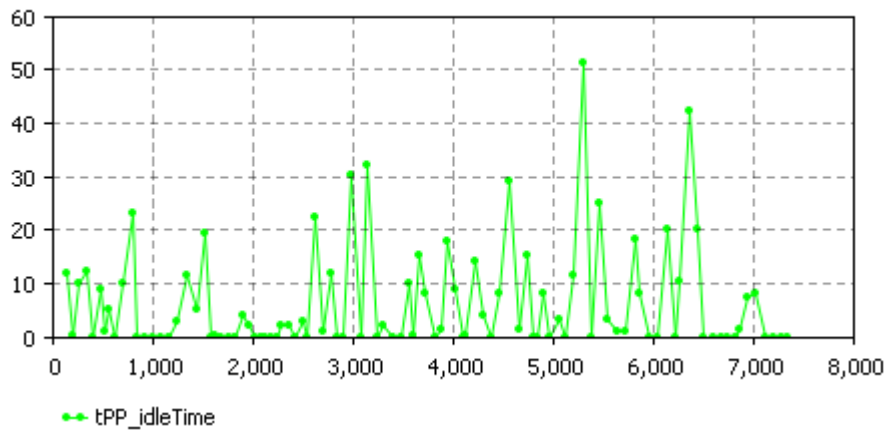


Figure A.137: cellPP (100 parts, 5 fixtures)

100 parts, 10 fixtures:

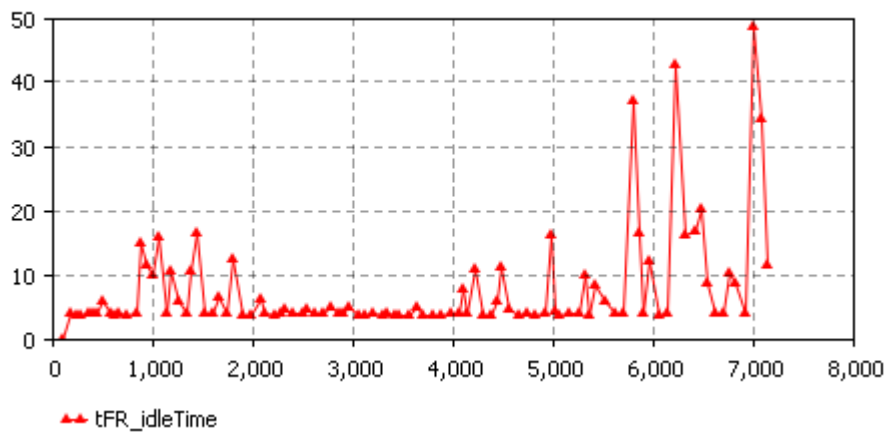


Figure A.138: cellFR (100 parts, 10 fixtures)

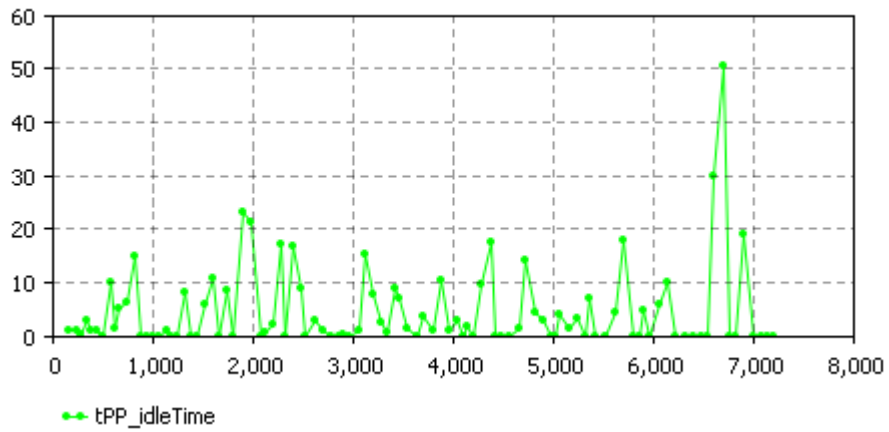


Figure A.139: cellPP (100 parts, 10 fixtures)

200 parts, 10 fixtures:

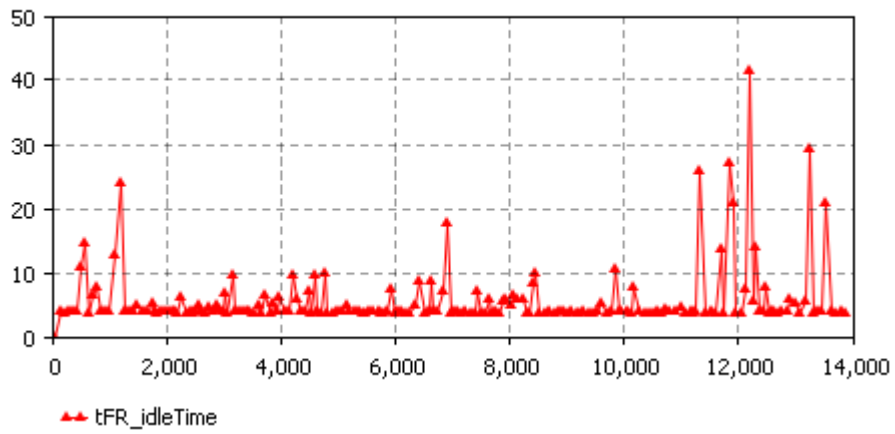


Figure A.140: cellFR (200 parts, 10 fixtures)

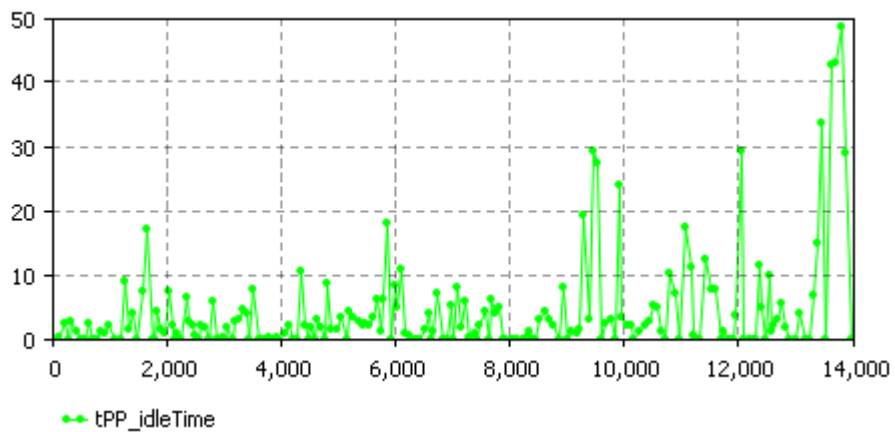


Figure A.141: cellPP (200 parts, 10 fixtures)

200 parts, 20 fixtures:

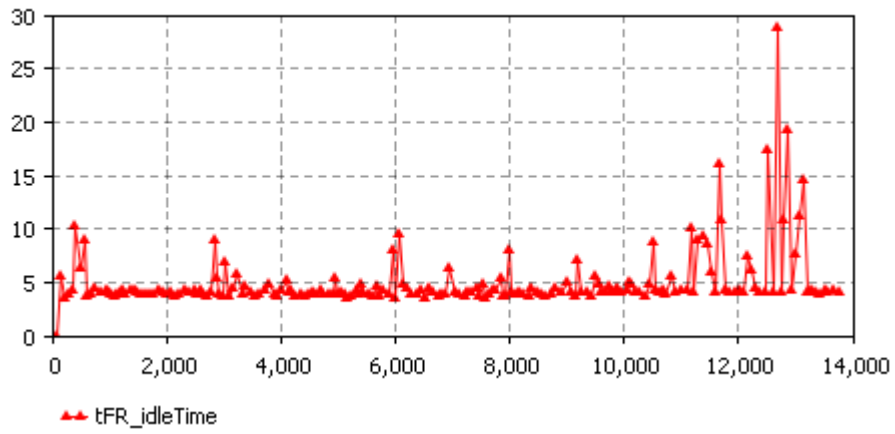


Figure A.142: cellFR (200 parts, 20 fixtures)

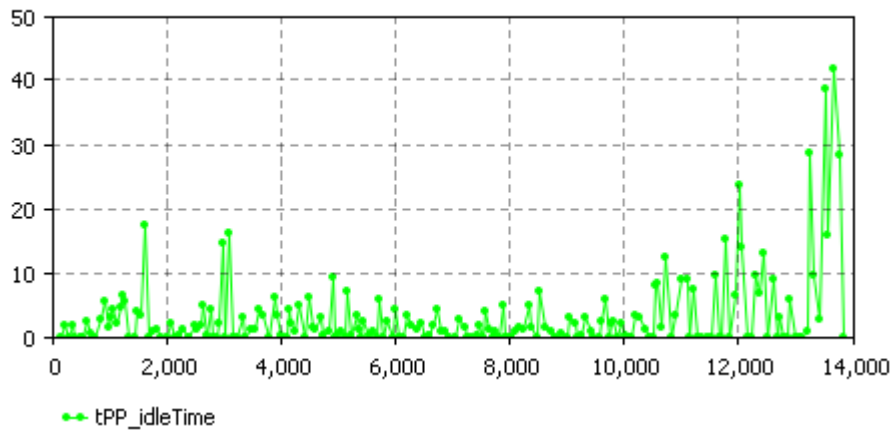


Figure A.143: cellPP (200 parts, 20 fixtures)

300 parts, 5 fixtures:

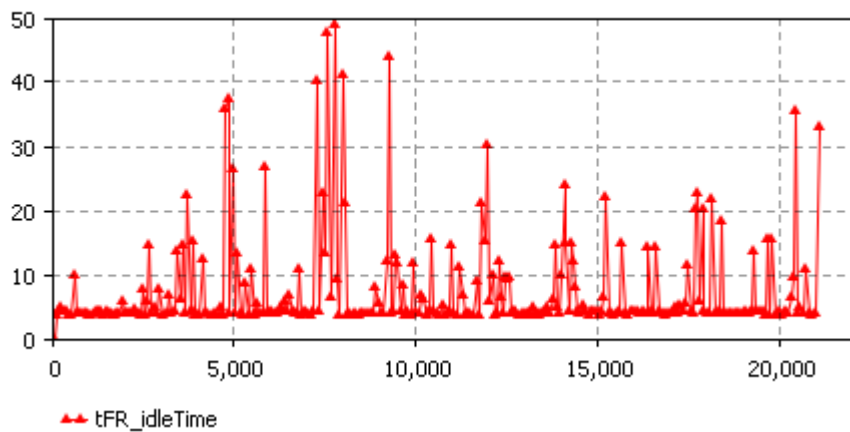


Figure A.144: cellFR (300 parts, 15 fixtures)

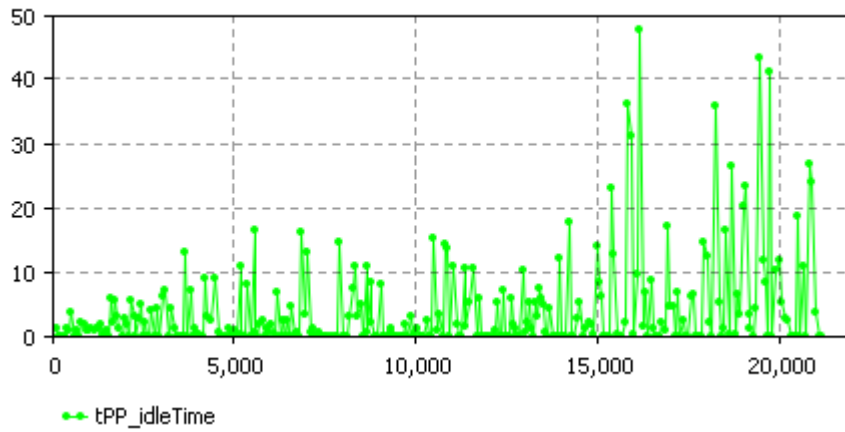


Figure A.145: cellPP (300 parts, 15 fixtures)

300 parts, 30 fixtures:

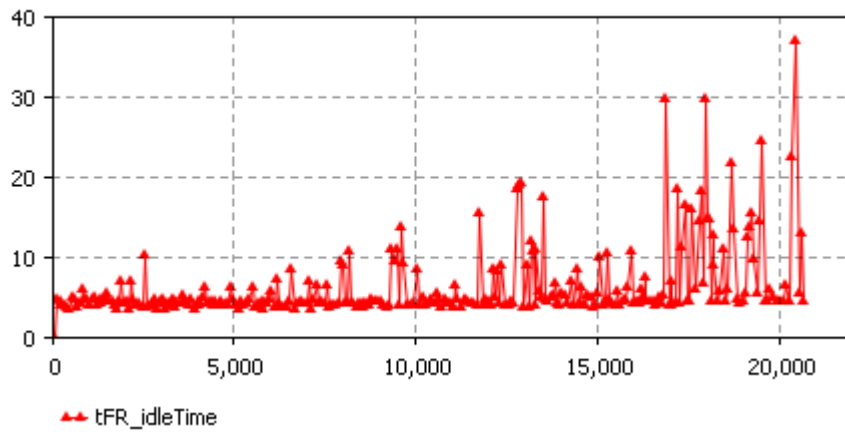


Figure A.146: cellFR (300 parts, 30 fixtures)

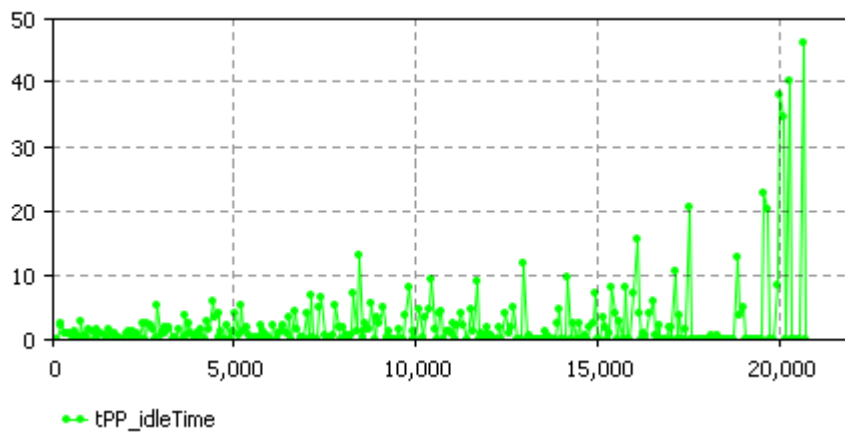


Figure A.147: cellPP (300 parts, 30 fixtures)

400 parts, 20 fixtures:

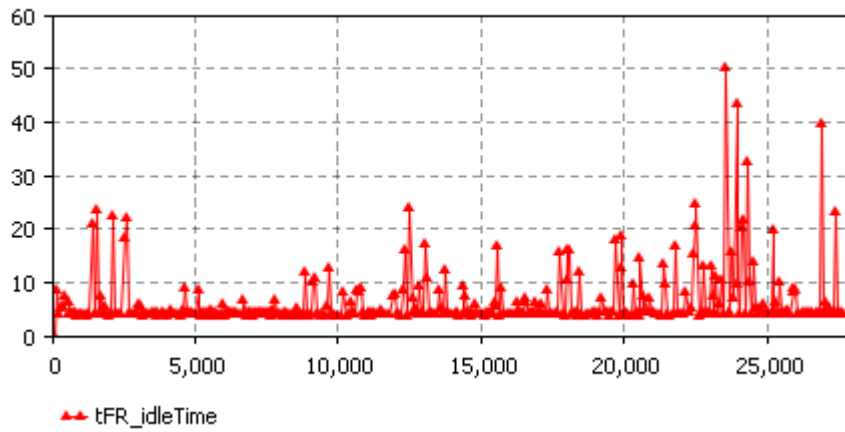


Figure A.148: cellFR (400 parts, 20 fixtures)

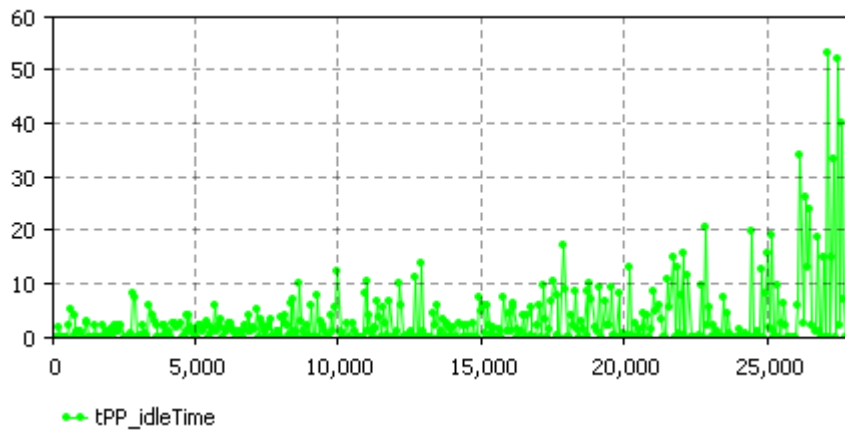


Figure A.149: cellPP (400 parts, 20 fixtures)

400 parts, 40 fixtures:

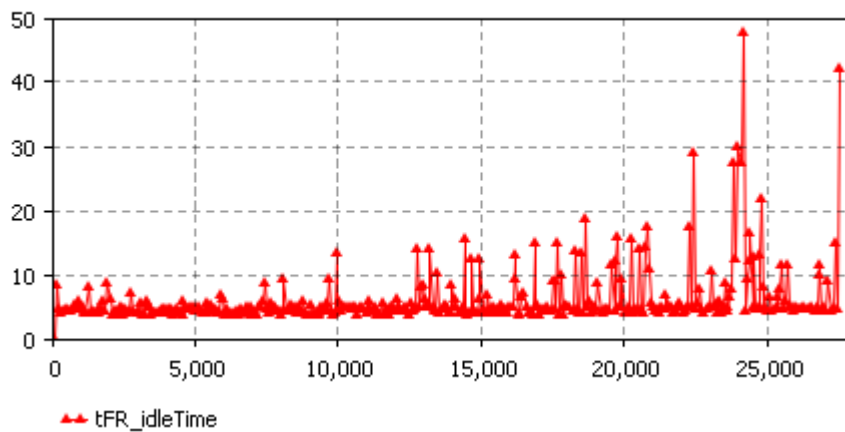


Figure A.150: cellFR (400 parts, 40 fixtures)

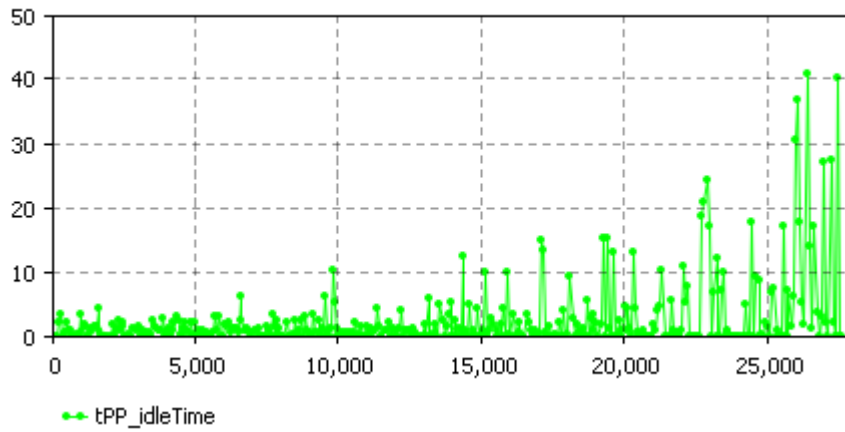


Figure A.151: cellPP (400 parts, 40 fixtures)

A.6.2. Utilisation Graphs

Figure A.152 to Figure A.167 present the utilisation graphs for the problems tested; Utilisation vs. Simulation Runtime (s) for every test. The graphs were separated for cellFR and cellPP for each test.

100 parts, 5 fixtures:

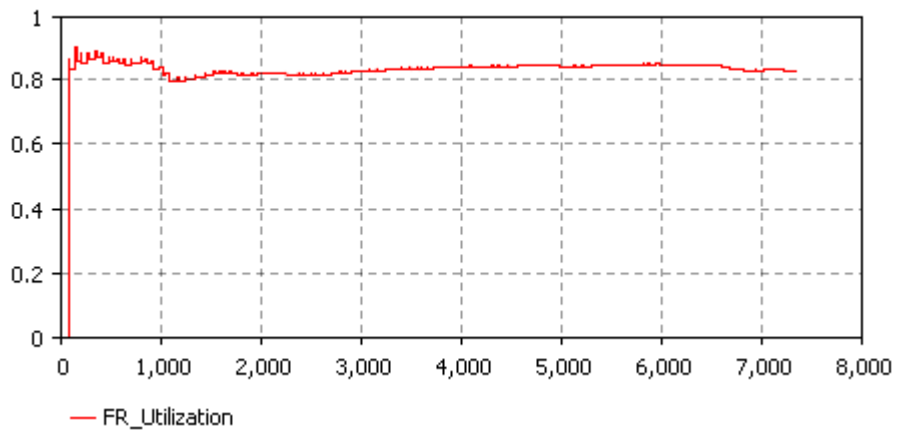


Figure A.152: cellFR (100 parts, 5 fixtures)

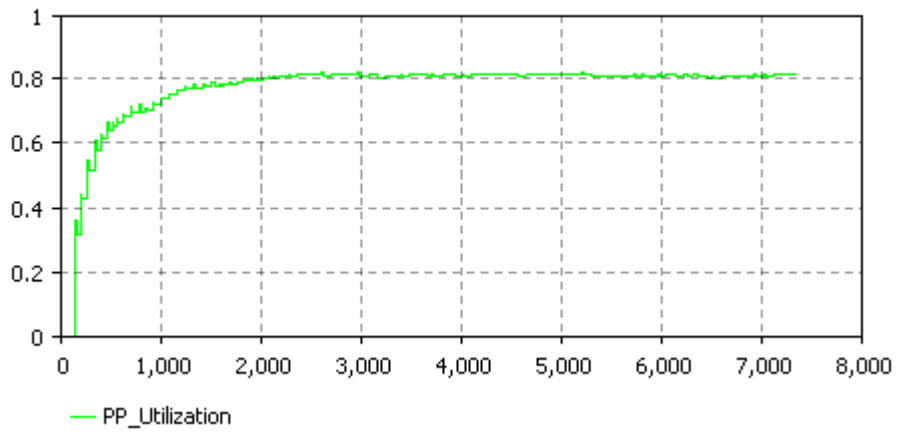


Figure A.153: cellPP (100 parts, 5 fixtures)

100 parts, 10 fixtures:

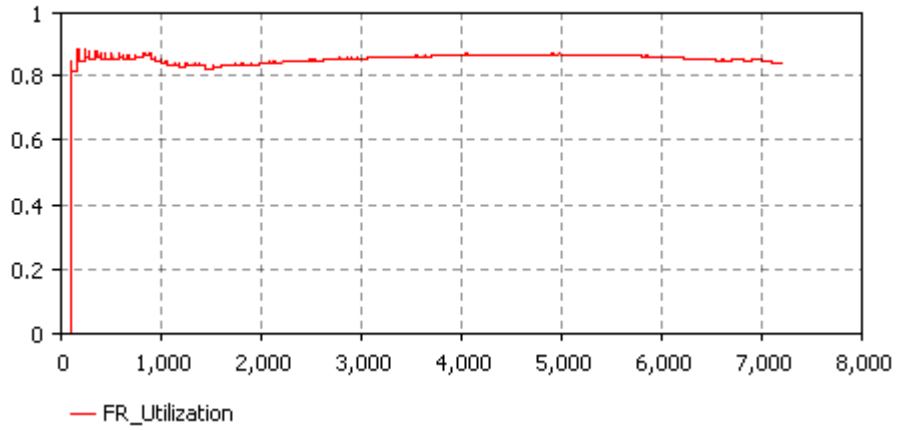


Figure A.154: cellFR (100 parts, 10 fixtures)

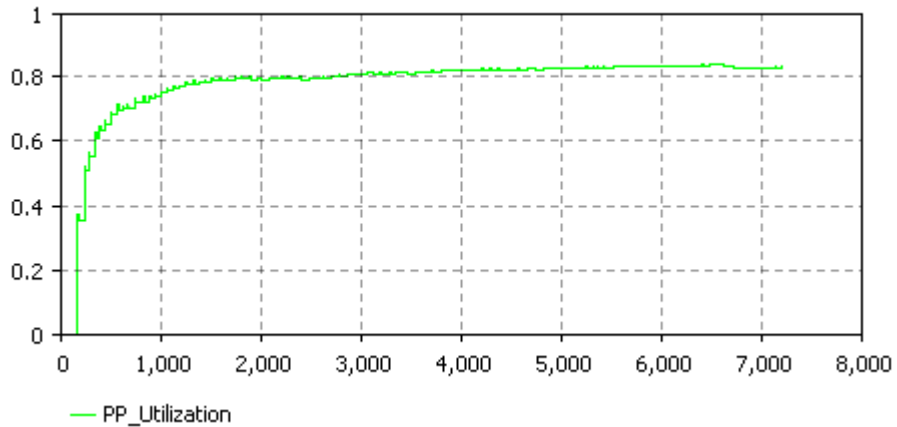


Figure A.155: cellPP (200 parts, 10 fixtures)

200 parts, 10 fixtures:

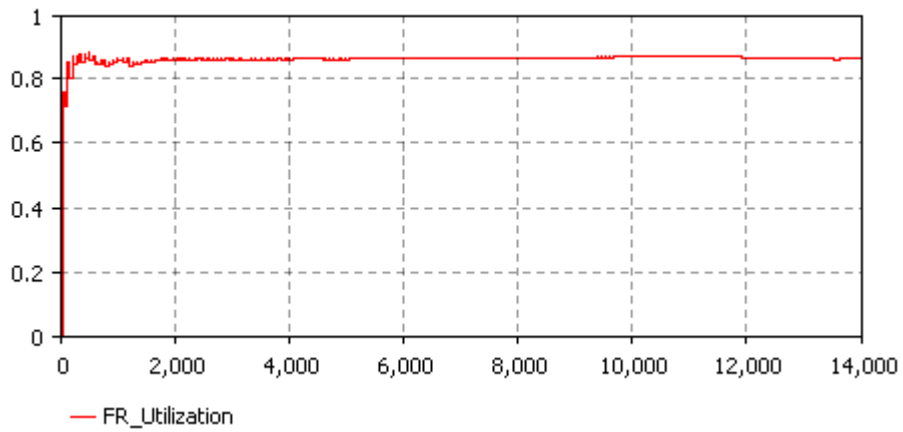


Figure A.156: cellFR (200 parts, 10 fixtures)

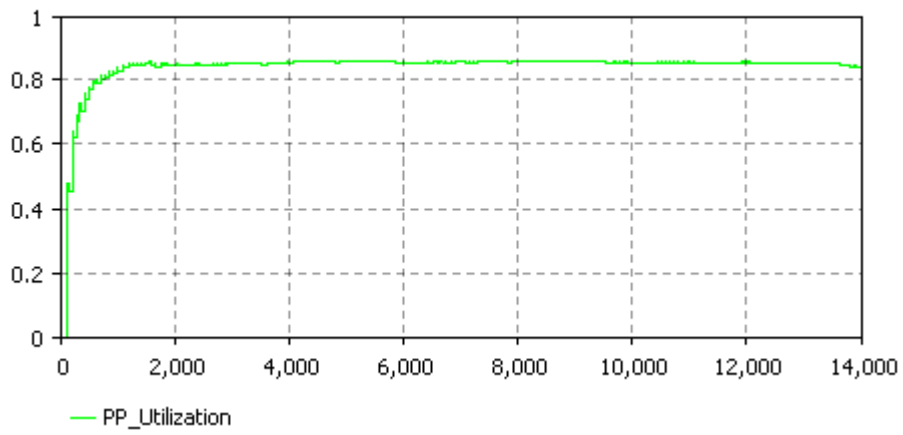


Figure A.157: cellPP (200 parts, 10 fixtures)

200 parts, 20 fixtures:

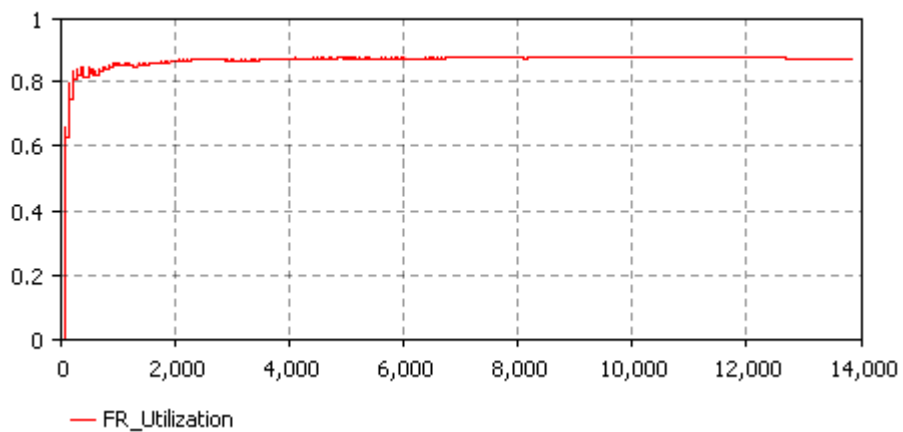


Figure A.158: cellFR (200 parts, 20 fixtures)

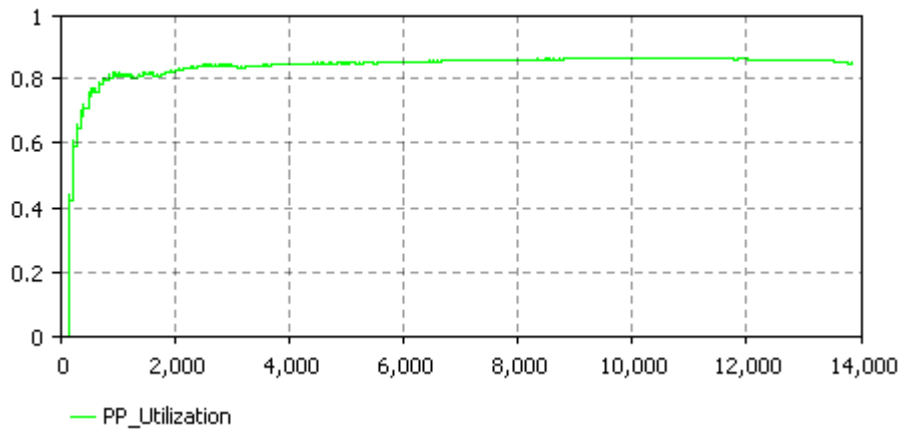


Figure A.159: cellPP (200 parts, 20 fixtures)

300 parts, 15 fixtures:

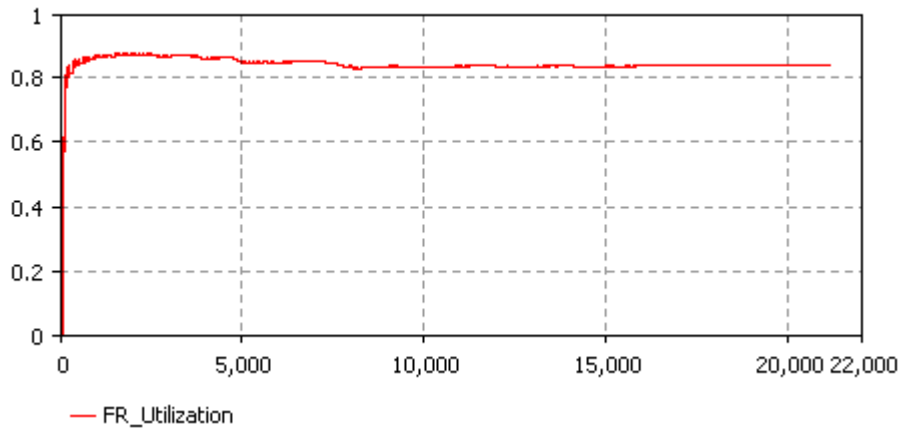


Figure A.160: cellFR (300 parts, 15 fixtures)

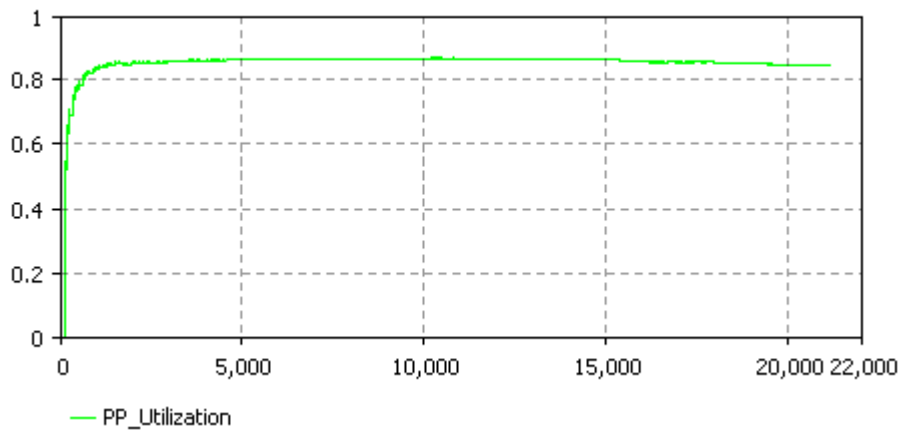


Figure A.161: cellPP (300 parts, 15 fixtures)

300 parts, 30 fixtures:

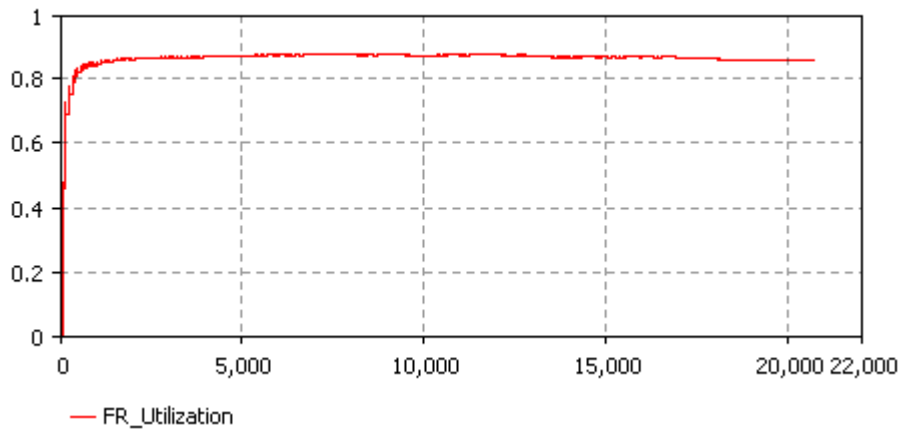


Figure A.162: cellFR (300 parts, 30 fixtures)

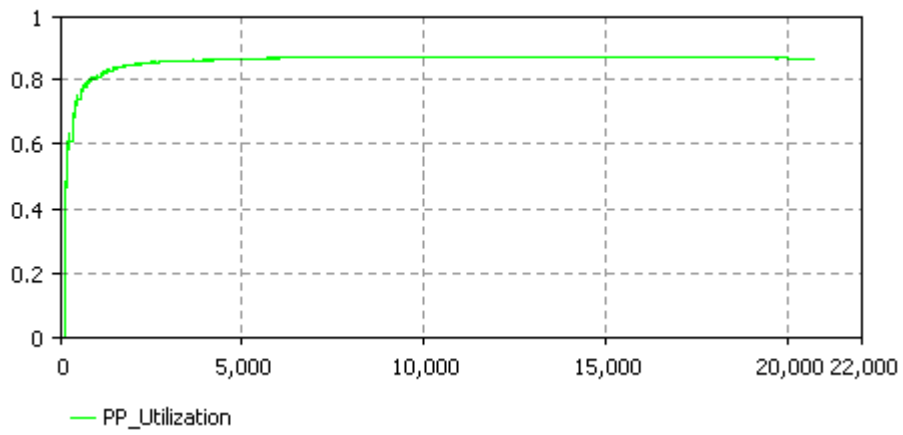


Figure A.163: cellPP (300 parts, 30 fixtures)

400 parts, 20 fixtures:

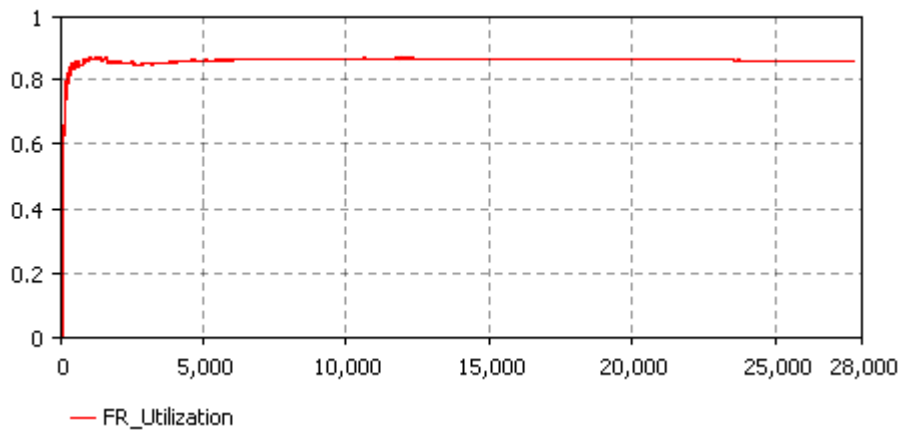


Figure A.164: cellFR (400 parts, 20 fixtures)

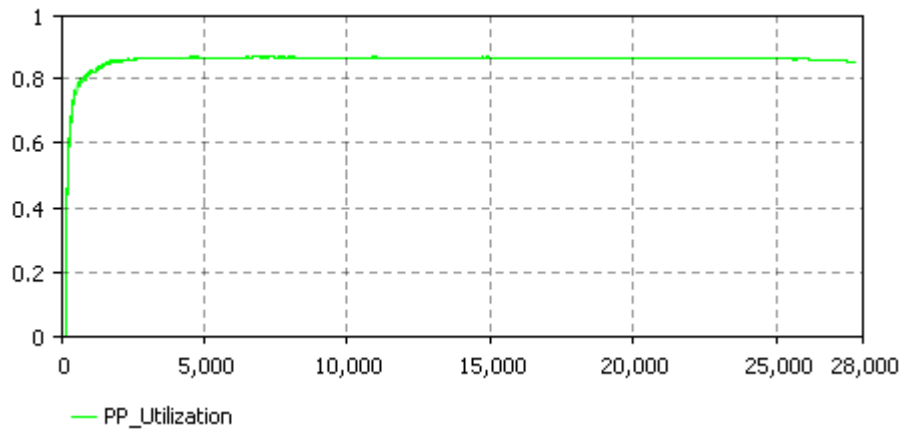


Figure A.165: cellPP (400 parts, 20 fixtures)

400 parts, 40 fixtures:

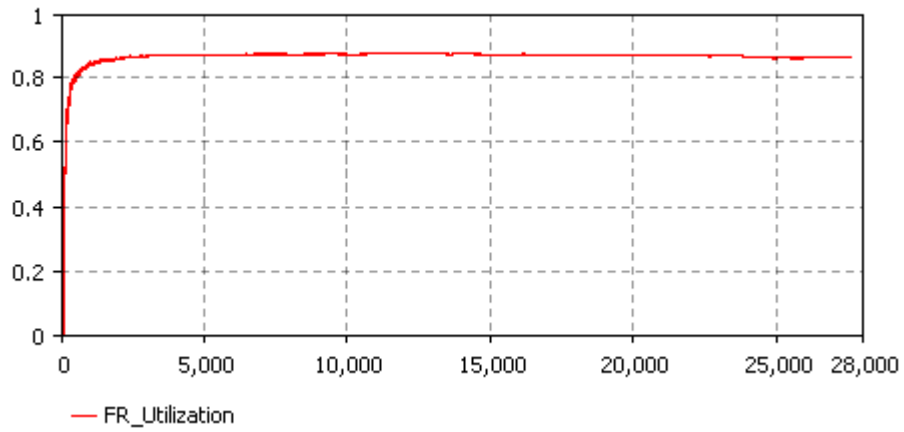


Figure A.166: cellFR (400 parts, 40 fixtures)

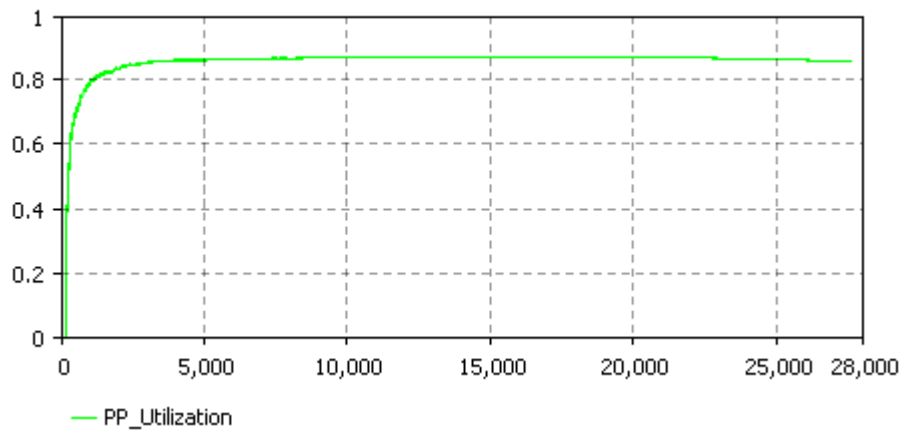


Figure A.167: cellPP (400 parts, 40 fixtures)

B. Appendix B: MATLAB Code

B.1. Clustering (Stage I and Stage II)

```
%% Parent function with main outputs
function [binaryConfigsList, configsComparisonList, distanceMatrix, stress,
unorderedClusters, silhouetteResults, fixtureSchedule, defaultOrder,
intraclusterImprovement, I, t_Clustering] = Clustering_08_Random(); %Parent
function
    tic
    configs = 6; %number of fixture configs for job list
    bases = 2; %number of fixture bases at disposal

    %fixture array dimensions
    dimRows = 8;
    dimCols = 8;
    dimSize = dimRows*dimCols;

    %for repeatable random number generation (RANDOM-specific)
    rng (1886,'twister');

    %active pin specifications (RANDOM-specific)
    minPins = 8; %at least 8 pins to fix part
    maxPins = 16; %at most 16 pins to fix part

    %similarity index coefficients
    ncA = 1;    ncB = 0;    ncC = 0;    ncD = 1;
    dcA = 1;    dcB = 1;    dcC = 1;    dcD = 1;

    %similarity index exponents
    neA = 1;    neB = 1.;    neC = 1;    neD = 1;
    deA = 1;    deB = 2;    deC = 2;    deD = 1;
    %Modified Rand index

    replicatesMDS = 100; %iterations for multi-dimensional scaling
    replicatesK = configs*(bases^2); %iterations for k-means clustering algorithm

    %% Local functions
    [binaryConfigsList] = initialiseBinaryConfigs(configs, dimRows, dimCols,
dimSize, minPins, maxPins);
    [configsComparisonList, pairwiseRow, distanceMatrix] =
configsComparisons(configs, binaryConfigsList, ncA, ncB, ncC, ncD, dcA, dcB, dcC,
dcD, neA, neB, neC, neD, deA, deB, deC, deD);
    [MDS_co_ords, stress] = MDS(pairwiseRow, replicatesMDS);
    [clusterIndices, centroids, intraDistSum,
interDist, clusterSize, maxSize, maxCluster, otherSizes, clusterDeficit, candidatesList, c
lusterKey, s, h, unorderedClusters, silhouetteResults] =
k_means(MDS_co_ords, bases, replicatesK, configs);

    [pR, New, Total, intraclusterRows, clusterColumn, pivot, remainder, fixtureSchedule, Link, d
efaultOrder, intraclusterImprovement, leafOrder, hDef, tDef, permDef, hOpt, tOpt, permOpt, s
q, ini, def, opt, imp1, imp2, imp3] =
intraclusterSequence(pairwiseRow, bases, unorderedClusters);
    [I] = matrixI(bases, maxSize, fixtureSchedule);
    t_Clustering = toc
end

%% 1. Generate random binary matrices to represent fixture configurations (RANDOM-
specific)
function [binaryConfigsList] = initialiseBinaryConfigs(configs, dimRows, dimCols,
dimSize, minPins, maxPins)

    binaryConfigsList = cell(configs, 2);
```

```

count = 1;

for i = 1:configs
    binaryConfigsList{count, 1} = sprintf('Config %d', i);

    binaryConfig = [zeros(1,dimSize-maxPins) ones(1,minPins) randi([0
1],1,maxPins-minPins)]; %constructs a binary string of 64, with at least 8 ones and
at most 16 ones (the final 8 bits are randomised).
    binaryConfig = binaryConfig(randperm(dimSize)); %creates random permutation
of above row vector
    binaryConfig = reshape(binaryConfig,[dimRows,dimCols]); %reshapes vector
into matrix of fixture dimensions (8x8)

    binaryConfigsList{count, 2} = binaryConfig;

    count = count + 1;

end

end

%% 2. Compare each fixture configuration to the other, find the a, b, c, d
parameters, and calc similarity index for each comparison
function [configsComparisonList, pairwiseRow, distanceMatrix] =
configsComparisons(configs, binaryConfigsList, ncA, ncB, ncC, ncD, dcA, dcB, dcC,
dcD, neA, neB, neC, neD, deA, deB, deC, deD)

    configsComparisonList = cell(3, 3);
    pairwiseRow = zeros(1, ((configs)^2-configs)/2);

    count = 1;

    configCompare = @(From, To) (From+To)./(From./To); %function for bsxfun to
compare fixture reconfigurations

    for i = 1:configs
        for j = i+1:configs

            configsComparisonList{count, 1} = sprintf('Config %d to %d', i, j);

            From = binaryConfigsList{i, 2}; %fixture config being changed from
            To = binaryConfigsList{j, 2}; %fixture config being changed to

            reconfigRelationship = bsxfun(configCompare,From,To); %create
comparison matrix for reconfiguration
            configsComparisonList{count, 2} = reconfigRelationship;

            %Similarity index parameters
            a = numel(find(reconfigRelationship==2)); %positive matches
            b = numel(find(reconfigRelationship==Inf)); %mismatches with positive
in To
            c = numel(find(reconfigRelationship==0)); %mismatches with positive in
From
            d = numel(find((isnan(reconfigRelationship)==1))); %negative matches

            %Similarity index equation
            S = ((ncA*a)^neA + (ncB*b)^neB + (ncC*c)^neC +
(ncD*d)^neD)/((dcA*a)^deA + (dcB*b)^deB + (dcC*c)^deC + (dcD*d)^deD); %modular form
of binary similarity measure

            configsComparisonList{count, 3} = S;

            %Pairwise dissimilarity row vector
            pairwiseRow(1, count) = 1-S; %conversion to dissimilarity measure (so
that higher similarities are closer together for the algorithm to cluster them)
            distanceMatrix = squareform(pairwiseRow);

```

```

        count = count + 1;

    end
end

end

%% 3. Multi-dimensional scaling - maps dissimilarities to a 2D plane
function [MDS_co_ords, stress] = MDS(pairwiseRow, replicatesMDS)
    %% 3.1. mdscale
    opts_MDS = statset('Display','final',...
        'MaxIter', Inf, ...
        'TolFun', 1e-6, ...
        'TolX', 1e-6);

    [MDS_co_ords, stress] = mdscale(pairwiseRow, 2, ... %scale pairwiseRow to TWO
dimensions
                                'criterion','stress',... %Kruskal's normalised
stress criterion for Stress1 equation used (non-metric MDS)
                                'Start','random',... %randomised start point
                                'Replicates', replicatesMDS, ... %number of re-
runs of algorithm
                                'Options', opts_MDS);

    %% 3.2. MDS plot
    figure('Name','Euclidean Plots');
    a = subplot(1,2,1);
    plot(MDS_co_ords(:,1), MDS_co_ords(:,2), '.', 'LineWidth', 2, 'MarkerSize', 36);
    daspect([1 1 1]); %equal axis spacing
    labels = num2str((1:size(MDS_co_ords,1))', '%d');
    text(MDS_co_ords(:,1), MDS_co_ords(:,2), labels, 'horizontal', 'left',
'vertical', 'bottom', 'FontSize', 16, 'FontWeight', 'bold'); %part numbers
    set(gca, 'FontSize', 16); %for ticks
    title(a, 'MDS output', 'FontSize', 30)
    xlabel('Scaled X Distance', 'FontSize', 30);
    ylabel('Scaled Y Distance', 'FontSize', 30);
    hold on
end

%% 4. K-means clustering
function [clusterIndices, centroids, intraDistSum,
interDist, clusterSize, maxSize, maxCluster, otherSizes, clusterDeficit, candidatesList, c
lusterKey, s, h, unorderedClusters, silhouetteResults] =
k_means(MDS_co_ords, bases, replicatesK, configs)

    %% 4.1. kmeans
    opts_kmeans = statset('Display','final');
    [clusterIndices, centroids, intraDistSum, interDist] =
kmeans(MDS_co_ords, bases, ...

'Distance','sqeuclidean',... %squared Euclidean distances used

'EmptyAction','drop',... %if cluster is empty, give it the furthest point from
another cluster centroid

'OnlinePhase','on',... %ensures local minimum of distance criterion is found
                                'MaxIter', Inf, ...

'Options', opts_kmeans, ...

'Replicates', replicatesK, ... %number of re-runs of algorithm
                                'Start','plus');
%uses k-means++ algorithm to initialise the centroid locations

    %% 4.2. Cluster size infeasibility fail-safe - if a cluster is so large, that
fixture recirculation would be infeasible in Stage III
    clusterSize = zeros(bases, 1);
    for i=1:bases

```

```

clusterSize(i,1) = sum(clusterIndices(:) == i); %size of each cluster
end

maxSize = max(clusterSize); %largest cluster size
[maxCluster] = find(clusterSize==maxSize); %largest cluster's index
otherSizes = sum(clusterSize)-maxSize; %sum of sizes of other (non-largest)
clusters
clusterDeficit = maxSize - otherSizes; %discrepancy

candidatesList = zeros([],bases);
%
if clusterDeficit > 1 %would render clusters infeasible for Stage III as-is
    maxPoints = find(clusterIndices==maxCluster); %list of which points belong
to largest cluster
    count = 1;
    for i=(maxPoints)'
        for j=[1:maxCluster-1 maxCluster+1:length(centroids)] %non-largest
cluster centroids
            candidatesList(count,j) = norm(MDS_co_ords(i,:)-centroids(j,:));
%distance from points in largest cluster to every other centroid; columns rep which
centroid
        end
        count = count + 1;
    end

    candidatesOrder = zeros(size(candidatesList));
    candidatesIndices = zeros(size(candidatesList));
    for i=[1:maxCluster-1 maxCluster+1:length(centroids)] %non-largest cluster
centroids
[candidatesOrder(:,i),candidatesIndices(:,i)]=sort(candidatesList(:,i),'ascend');
%creates two lists of min distance to largest for every cluster (min for each point
in largest cluster to any centroid), and the indices of these wrt candidatesList
    end
    candidatesOrder(~candidatesOrder)=Inf; %inserts Inf for largest cluster's
values
    candidatesIndices(~candidatesIndices)=Inf;

    A=[];
    B=[];
    count=1;
    for i=1:clusterDeficit-1 %to reduce deficit to 1, making Stage III feasible
        A(i,:) = (candidatesOrder(i,:)); %wanted points as per deficit
        [distance,index]=min(A(i,:));
        B(count,:)= [distance,index]; %extracting the actual minimum distance
(to that nearest centroid) and the index for candidatesList
        count=count+1;
    end

    for i=1:clusterDeficit-1
        clusterIndices(maxPoints(candidatesIndices(i,B(i,2))),1) = B(i,2);
%reassigns nearest point to another cluster in largest cluster to that nearest
cluster
    end
    Note = sprintf('Fail-safe heuristic used')
end

%Recalculating new key cluster values
clusterSize = zeros(bases, 1);
for i=1:bases
clusterSize(i,1) = sum(clusterIndices(:) == i);
end
maxSize = max(clusterSize);
%}
%% 4.3. K-means plot (RANDOM-specific)
b = subplot(2,2,[2 4]);
colours = ['b','g','r','c','m','y'];
markers = ['o','+','*','s','d','p'];

```

```

opts_CM = 6; %number of colour/marker options
for i=1:bases
    x = ceil(i/opts_CM)-1; %used as manipulator for cycling through options
    col = (i-x*6+x) - (ceil((i-x*6+x)/6)-1)*6; %ensures colour count runs in
cycles of 6, but shifts one up whenever marker is changed
    mark = i-x*6; %ensures marker count runs in cycles of 6 as i increases
    plot(MDS_co_ords(clusterIndices==i,1), MDS_co_ords(clusterIndices==i,2),
[colours(col), markers(mark)], 'MarkerSize',12, 'LineWidth',2);
    hold on

end

plot(centroids(:,1), centroids(:,2), 'kx', 'MarkerSize',20, 'LineWidth',4);
%centroid X's
daspect([1 1 1]); %equal axis spacing
clusterKey = cell(1,bases);
for i = 1:bases
    clusterKey{i} = sprintf('Cluster %d', i);
end
legend(clusterKey{1:bases}, 'Centroid', 'Location', 'NW');
set(gca, 'FontSize',16) %for ticks and legend
title (b, 'K-means clustering output', 'FontSize',30);
xlabel('Scaled X Distance', 'FontSize',30);
ylabel('Scaled Y Distance', 'FontSize',30);
hold on

%% 4.4. Recording clusters
unorderedClusters = cell(bases, 2);
for i = 1:bases
    unorderedClusters{i, 1} = sprintf('Fixture base %d', i);
    unorderedClusters{i, 2} = find(clusterIndices==i); %extracts the data
points belonging to each cluster i
end

%% 4.5. Silhouette plot
figure('Name', 'Silhouette Plot');
hold on
[s,h] = silhouette(MDS_co_ords,clusterIndices); %create silhouette plot
set(gca, 'FontSize',16, 'XLim', [-1 1]); %for ticks and range
title('Silhouette Plot', 'FontSize',30);
xlabel('Silhouette Value', 'FontSize',30);
ylabel('Cluster', 'FontSize',30);
hold on

%% 4.6. Recording silhouette values
sR = cell([],2);
silhouetteResults = cell(configs+bases,2);
count = 1;

for i=1:bases
    a = unorderedClusters{i,2};
    a = s(a);
    [x, y] = intersect(s,a);
    for j=1:length(a)
        sR{j,1} = y(j);
        sR{j,2} = x(j);
    end
    [~, y] = sort([sR{:,2}], 'descend');
    sR = sR(y,:);
    silhouetteResults{count,1} = sprintf('Cluster');
    silhouetteResults{count,2} = sprintf('%d', i);
    silhouetteResults(count+1:count+j, 1:2) = sR(1:j, 1:2);
    count = count + j + 1;
end
end

%% 5. Intracluster sequencing
function
[pR, New, Total, intraclusterRows, clusterColumn, pivot, remainder, fixtureSchedule, Link, d

```

```

efaultOrder,intraclusterImprovement,leafOrder,hDef,tDef,permDef,hOpt,tOpt,permOpt,s
q,ini,def,opt,imp1,imp2,imp3] =
intraclusterSequence(pairwiseRow,bases,unorderedClusters)

    %% 5.1. Synthesising data for intracluster sequencing
    pR = squareform(pairwiseRow); %creates comparison matrix from pairwise row of
dissimilarities
    New = zeros(1,[]);
    Total = zeros(1,[]);
    intraclusterRows = cell(bases, 1); %pairwise distance row vector for each
cluster

    for i = 1:bases
        for j=1:length(unorderedClusters{i, 2}) %counts through members of cluster
i
            clusterColumn = unorderedClusters{i,2}; %indices of cluster number i
pivot = clusterColumn(j,1); %the row to explore in pR comparison matrix
            if length(clusterColumn) > 1
                remainder = clusterColumn(j+1:length(clusterColumn))'; %row vector
formed by comparing the similarities between the pivot all members in clusterColumn
that follow it
                New(1,1:length(remainder)) = pR(pivot,remainder); %the columns to
explore in pR comparison matrix (i.e. members of clusterColumn after pivot);
converted to a row vector
                Total = horzcat(Total,New); %joining the row vectors formed thus
far to the one created in the current loop
                New = zeros(1,[]); %emptying New row vector for next iteration
(next pivot)
            else
                Total = pivot;
            end
        end
        intraclusterRows{i,1} = Total; %printing the accumulated row vector for
cluster i to the i-th row in intraclusterRows cell array
        Total = zeros(1,[]); %emptying Total row vector for next iteration (next
cluster)
    end

    %% 5.2. Optimal order for each cluster (minimum cumulative pairwise
dissimilarity)
    fixtureSchedule = cell(bases, 2);
    Link = zeros([],3);
    defaultOrder = cell(bases, 2);
    intraclusterImprovement = cell(bases, 4);

    for i = 1:bases
        fixtureSchedule{i, 1} = sprintf('Fixture base %d', i);
        defaultOrder{i, 1} = sprintf('Fixture base %d', i);

        if length(unorderedClusters{i,2}) > 1 %in case cluster has only one member
- no pairwise distance to compute
            Link = linkage(intraclusterRows{i,1}, 'single'); %links parts based on
closest similarity (single linkage) in heirachical order (i.e. closest pairs linked
first and subsequent closest points appended thereafter)
            leafOrder = optimalleaforder(Link, intraclusterRows{i,1}); %optimally
orders clusters such that the shortest distance (ito similarity measure) is
traversed
            fixtureSchedule{i, 2} = unorderedClusters{i,2}(leafOrder); %maps part
names to intracluster order

            %Plot each dendrogram
            str = sprintf('Fixture Base #%d', i);
            figure('Name', str);
            %Default leaf order
            subplot(2,1,1);
            [hDef,tDef,permDef] = dendrogram(Link); %output dendrogram order
permutation
            set(gca,'FontSize',16) %for ticks

```

```

        set(gca, 'XTickLabel', unorderedClusters{i,2}(permDef)); %map ticks to
part numbers
        set(gca, 'YLim', [0 1]); %range from 0 to 1 (inf for auto)
        title('Default Leaf Order', 'FontSize', 30);
        ylabel('Dissimilarity Value', 'FontSize', 20);
        xlabel('Part Sequence', 'FontSize', 20);
        hold on
        %Optimal leaf order
        subplot(2,1,2);
        [hOpt,tOpt,permOpt] = dendrogram(Link, 'reorder', leafOrder); %output
dendrogram order permutation - permOpt same as leafOrder
        set(gca, 'FontSize', 16)
        set(gca, 'XTickLabel', unorderedClusters{i,2}(permOpt)); %map ticks to
part numbers
        set(gca, 'YLim', [0 1]); %range from 0 to 1 (inf for auto)
        title('Optimal Leaf Order', 'FontSize', 30);
        ylabel('Dissimilarity Value', 'FontSize', 20);
        xlabel('Part Sequence', 'FontSize', 20);
        hold on

        defaultOrder{i, 2} = unorderedClusters{i,2}(permDef); %prints default
dendrogram order of parts for record

        %Calculating intracluster improvement
        intraclusterImprovement{i, 1} = sprintf('Fixture base %d', i);

        sq = squareform(intraclusterRows{i,1}); %distance matrix for cluster i

        ini = 0;
        def = 0;
        opt = 0;

        for j = 1:length(unorderedClusters{i,2})-1
            ini = ini + sq(j,j+1); %calculate distance as per initial cluster
order
        end
        for j = 1:length(permDef)-1
            def = def + sq(permDef(j),permDef(j+1)); %calculate distance as per
default dendrogram order
        end
        for j = 1:length(permOpt)-1
            opt = opt + sq(permOpt(j),permOpt(j+1)); %calculate distance as per
optimal dendrogram order
        end

        imp1 = ((ini-def)/ini) * 100; %percentage improvement of default over
initial for cluster i
        imp2 = ((def-opt)/def) * 100; %calculate percentage improvement of
optimal over default for cluster i
        imp3 = ((ini-opt)/ini) * 100; %calculate percentage improvement of
optimal over initial (total) for cluster i

        intraclusterImprovement{i, 2} = imp1;
        intraclusterImprovement{i, 3} = imp2;
        intraclusterImprovement{i, 4} = imp3;
        %}
    else
        fixtureSchedule{i, 2} = unorderedClusters{i,2}; %if only one object in
cluster, just write the object to fixtureSchedule as-is
    end

    Link = zeros([],3);
end
end

%% 6. I matrix for MILP model
function [I] = matrixI(bases,maxSize,fixtureSchedule)
    I = zeros(bases,maxSize);

```

```
for i=1:bases
    I(i,1:length(fixtureSchedule{i,2})) = fixtureSchedule{i,2};
end
end
```


B.2. MILP Model (Stage III)

```
function [f,intcon,A,b,Aeq,beq,lb,ub,variables,listR,listT,t_createVariables,K_tot]
= LP_test20(I,distanceMatrix); %Parent function

%%Instructions:
%1. Run Clustering_08 to obtain I and distanceMatrix.
%3. Copy and paste Line 1 from '[' to ';' in command window and press enter to
create variables.
%4. Copy and paste commented section below (from 'tic' to 't_Solution = toc') in
command window to solve problem and obtain results.

%{
tic

options = optimoptions('intlinprog',...
    'BranchRule','mostfractional',... %maxfun
    'ConstraintTolerance',1e-3,...
    'CutGeneration','none',...
    'Heuristics','rss',... %round %rins
    'HeuristicsMaxNodes',Inf,...
    'IntegerPreprocess','none',... %advanced
    'IntegerTolerance',1e-3,...
    'LPMaxIterations',Inf,...
    'LPOptimalityTolerance',1e-6,... %1e-1 %1e-3 %1e-9
    'MaxNodes',Inf,...
    'MaxTime',432000,... %(3_days)
    'NodeSelection','minobj',...
    'OutputFcn',@savemilpsolutions,...
    'PlotFcn',@optimplotmilp,...
    'ObjectiveImprovementThreshold',1e-3,...
    'RelativeGapTolerance',1e-3,...
    'RootLPAlgorithm','primal-simplex',...
    'RootLPMaxIterations',Inf);

[x fval exitflag output] = intlinprog(f,intcon,A,b,Aeq,beq,lb,ub,options);

exitflag
fval
output

for i=1:length(x)
variables{i,3} = x(i,1);
end

%%Results
Total_idleTime = fval;

Find = find(x);
activeVariables_List = cell(size(Find,1),3);

for i=1:size(Find,1)
    activeVariables_List{i,1} = variables{Find(i),1};
    activeVariables_List{i,2} = variables{Find(i),2};
    activeVariables_List{i,3} = variables{Find(i),3};
end

X_List = zeros(K_tot,3);
for i=1:K_tot
    X_List(i,:) = activeVariables_List{i,2};
end
X_List = sortrows(X_List,3);

Phi_List = zeros((K_tot-1),6);
count = 1;
for i=K_tot+1:K_tot+(K_tot-1)
    Phi_List(count,:) = activeVariables_List{i,2};
```

```

    count = count + 1;
end
Phi_List = sortrows(Phi_List,6);

%shows which pairs of fixture-part combinations are synchronised for the cells
syncPairs = cell((K_tot-1),2);
for i=1:K_tot-1
    syncPairs{i,1} = Phi_List(i,[2 4 6]);
    syncPairs{i,2} = Phi_List(i,[1 3 5]);
end

finalSchedule = cell(K_tot+1,2);
finalSchedule{1,1} = 'Fixture';
finalSchedule{1,2} = 'Part';

for i=1:K_tot-1
    finalSchedule{i+1,1} = Phi_List(i,1); %fixture used
    finalSchedule{i+1,2} = I(Phi_List(i,1),Phi_List(i,3)); %part number
    count = i+1

    if count == K_tot %for final part
        finalSchedule{count+1,1} = Phi_List(i,2);
        finalSchedule{count+1,2} = I(Phi_List(i,2),Phi_List(i,4));
    end
end

t_Solution = toc
%}

tic
%set of sets of parts in respective cluster order - rows rep fixtures, columns rep
intercluster order of parts, integers rep part names (irrelevant to solver); zero
for empty slot:

%I generated from Clustering_08

I_tot = size(I,1); %number of bases - rows in I

J = sum(I~=0,2); %number of parts per fixture; row reps fixture, element reps
number of parts; calc as number of non-zero elements in each row
J_tot = sum(J); %total number of parts
%combos of J; number of valid pairs of fixture-part combinations, i.e. j-j_alt
combos:
J_combos = nchoosek(J,2); %pairs of J combos
J_combos_tot = sum(J_combos(:,1).*J_combos(:,2))*2; %sum of J combos

K_tot = nnz(I); %number of jobs; non-zero elements in I

X_tot = J_tot*K_tot; %number of X_ijk variables
Phi_tot = J_combos_tot*(K_tot-1); %number of variables for each
Phi_i,i_alt,j,j_alt,k,k_alt; reps pairs, so counted up to (K_tot-1) only.
Omega_tot = Phi_tot; %Zeta defined by same set as Phi

nVars = X_tot + Phi_tot + Omega_tot; %total number of variables

intcon = 1:X_tot; %which variables are integers

rng(1886, 'twister'); %for repeatable random number generation

%Create random reconfiguration times for Cell 1
countR = 1;
listR = cell(J_tot,3);

for i=1:I_tot
    for j=1:J(i,1)
        listR{countR, 1} = 'R';
        listR{countR, 2} = [i,j]; %for each fixture changing from previous config
to required config

```

```

        if j>1
            listR{countR, 3} = 30 + ceil(distanceMatrix(I(i,j),I(i,(j-1)))*60);
%time values between range of 30-90 seconds, related to dissimilarity value
        else
            listR{countR, 3} = randi([30 90],1,1); %random time values between
range of 30-90 seconds
        end
        countR = countR + 1;
    end
end

%Create random processing times for Cell 2
countT = 1;
listT = cell(J_tot,3);
for i=1:I_tot
    for j=1:J(i,1)
        listT{countT,1} = 'T';
        listT{countT,2} = [i,j]; %for which ij pair
        listT{countT,3} = randi([30 90],1,1); %random time values between range of
30-90 seconds
        countT = countT + 1;
    end
end

[variables] = identifyDecisionVariables(I_tot,J,K_tot,nVars); %Local function 1
[f,A,b,Aeq,beq,lb,ub] =
pop_Matrices(I_tot,J,J_tot,K_tot,X_tot,Phi_tot,Omega_tot,nVars,variables,listT,list
R); %Local function 2
t_createVariables = toc
end

%% identifyDecisionVariables function
function [variables] = identifyDecisionVariables(I_tot,J,K_tot,nVars) %Local
function 1 %creates list of decision variables to neatly sort data

prefixList=['X','P','O']; %Matrix of strings to separate variables

variables=cell(nVars,3); %Cell array to sort data

%%Create X variables
var_Counter=1;
for i=1:I_tot
    for j=1:J(i,1) %for each j related to i
        for k=1:K_tot
            indices=[i,j,k];
            variables{var_Counter,1}= prefixList(1); %Prints 'X'
            variables{var_Counter,2}= indices;
            var_Counter=var_Counter+1;
        end
    end
end

%%Create Phi variables
for i=1:I_tot
    for i_alt=1:I_tot
        if i_alt ~= i %another fixture base
            for j=1:J(i,1)
                for j_alt=1:J(i_alt,1) %a part from that other fixture base
                    for k=1:K_tot-1
                        k_alt = k + 1;
                        indices=[i,i_alt,j,j_alt,k,k_alt]; %resets indices for Phi
variable
                        variables{var_Counter,1}= prefixList(2); %Prints 'P' for
Phi
                        variables{var_Counter,2}= indices;
                        var_Counter=var_Counter+1; %count continues from X loop
                    end
                end
            end
        end
    end
end
end

```

```

        end
    end
end
end

%Create Omega variables
for i=1:I_tot
    for i_alt=1:I_tot
        if i_alt ~= i %another fixture base
            for j=1:J(i,1)
                for j_alt=1:J(i_alt,1) %a part from that other fixture base
                    for k=1:K_tot-1 %number of fixture-part combination pairs
required
                        k_alt = k + 1;
variable
                        indices=[i,i_alt,j,j_alt,k,k_alt]; %resets indices for Zeta
Omega
                        variables{var_Counter,1}= prefixList(3); %Prints 'O' for
                        variables{var_Counter,2}= indices;
                        var_Counter=var_Counter+1; %count continues from Phi loop
                    end
                end
            end
        end
    end
end
end
end

end

%% pop_Matrices function
function [f,A,b,Aeq,beq,lb,ub] =
pop_Matrices(I_tot,J,J_tot,K_tot,X_tot,Phi_tot,Omega_tot,nVars,variables,listT,list
R) %Local function 2 %Populates coefficients of matrices A,b,Aeq,beq,lb,ub and f
for solver

A=zeros([],nVars); %Creates "A" matrix with rows equivalent to constraints and
columns equivalent decision variables
b=zeros([],1); %Creates column vector "b" with rows equiv to constraints
A=sparse(A); %sparse for minimised file size
b=sparse(b);

Aeq=zeros([],nVars); %Like A, but for equations instead of inequalities
beq=zeros([],1);
Aeq=sparse(Aeq);
beq=sparse(beq);

row_Counter = 1; %counter for A and b
row_eqCounter = 1; %counter for Aeq and beq

%populate idle-time comparison constraint (T-R's for valid X's) - Phi
for i=1:I_tot
    for i_alt=1:I_tot
        if i_alt~=i
            for j=1:J(i,1)
                for j_alt=1:J(i_alt,1)
                    for k=1:K_tot-1
                        k_alt = k + 1; %because k_alt is always for the subsequent
time period

                                %Phi linearisation sub-constraint 1 (+obj fcn)
                                [col_posPhi] =
findVariablePhi(variables,X_tot,nVars,'P',i,i_alt,j,j_alt,k,k_alt); %searches list
of variables for position of Phi variable on each run
                                A(row_Counter,col_posPhi) = -1; %prints value for that Phi
variable to its position in A matrix on each run
                                b(row_Counter,1) = 0; %prints value for that Phi variable
to its position in b column vector on each run

```

```

        [col_posOmega] =
findVariableOmega(variables,X_tot,Phi_tot,Omega_tot,'O',i,i_alt,j,j_alt,k,k_alt);
[T] = findVariableT(listT,J_tot,'T',i,j);
[R] = findVariableR(listR,J_tot,'R',i_alt,j_alt);
A(row_Counter,col_posOmega) = (1*listT{T,3} -
1*listR{R,3}); %T_ij - R_iAlt,jAlt (i.e. T-R) for that Omega

        row_Counter = row_Counter + 1;

        %Phi linearisation sub-constraint 2 (-obj fcn)
        [col_posPhi] =
findVariablePhi(variables,X_tot,nVars,'P',i,i_alt,j,j_alt,k,k_alt);
A(row_Counter,col_posPhi) = -1;
b(row_Counter,1) = 0;

        [col_posOmega] =
findVariableOmega(variables,X_tot,Phi_tot,Omega_tot,'O',i,i_alt,j,j_alt,k,k_alt);
[T] = findVariableT(listT,J_tot,'T',i,j);
[R] = findVariableR(listR,J_tot,'R',i_alt,j_alt);
A(row_Counter,col_posOmega) = -(1*listT{T,3} -
1*listR{R,3}); %T_ij - R_iAlt,jAlt (i.e. T-R) for that Omega

        row_Counter = row_Counter + 1;

    end
end
end
end
end

%populate synchronous constraint (both X's valid for every time period) - Omegas
for i=1:I_tot
    for i_alt=1:I_tot
        if i_alt~=i
            for j=1:J(i,1)
                for j_alt=1:J(i_alt,1)
                    for k=1:K_tot-1
                        k_alt = k + 1;
                        k_next = k + 1; %for reconfig in next time period

                        %Omega linearisation sub-constraint 1
                        [col_posOmega] =
findVariableOmega(variables,X_tot,Phi_tot,Omega_tot,'O',i,i_alt,j,j_alt,k,k_alt);
A(row_Counter,col_posOmega) = -1;
b(row_Counter,1) = 1;

                        [col_posX0,~,~] =
findVariableX(variables,X_tot,'X',i,[],j,[],k,[],[]);
A(row_Counter,col_posX0) = 1;

                        [~,col_posX1,~] =
findVariableX(variables,X_tot,'X',[],i_alt,[],j_alt,[],[],k_next); %finds a j for
another i for the next time period
                        A(row_Counter,col_posX1) = 1;

                        row_Counter = row_Counter + 1;

                        %Omega linearisation sub-constraint 2
                        [col_posOmega] =
findVariableOmega(variables,X_tot,Phi_tot,Omega_tot,'O',i,i_alt,j,j_alt,k,k_alt);
A(row_Counter,col_posOmega) = 1;
b(row_Counter,1) = 0;

                        [col_posX0,~,~] =
findVariableX(variables,X_tot,'X',i,[],j,[],k,[],[]);
A(row_Counter,col_posX0) = -1;

```

```

        row_Counter = row_Counter + 1;

        %Omega linearisation sub-constraint 3
        [col_posOmega] =
findVariableOmega(variables,X_tot,Phi_tot,Omega_tot,'O',i,i_alt,j,j_alt,k,k_alt);
        A(row_Counter,col_posOmega) = 1;
        b(row_Counter,1) = 0;

        [~,col_posX1,~] =
findVariableX(variables,X_tot,'X',[],i_alt,[],j_alt,[],[],k_next);
        A(row_Counter,col_posX1) = -1;

        row_Counter = row_Counter + 1;

    end
end
end
end
end

%populate required number of valid Omegas to exist
for i=1:I_tot
    for i_alt=1:I_tot
        if i_alt~=i
            for j=1:J(i,1)
                for j_alt=1:J(i_alt,1)
                    for k=1:K_tot-1
                        k_alt = k + 1;
                        [col_posOmega] =
findVariableOmega(variables,X_tot,Phi_tot,Omega_tot,'O',i,i_alt,j,j_alt,k,k_alt);
                        Aeq(row_eqCounter,col_posOmega) = 1;
                        beq(row_eqCounter,1) = K_tot-1; %number of fixture-part
combination pairs required
                    end
                end
            end
        end
    end
end
end
row_eqCounter = row_eqCounter + 1; %placed here because above loop is a summation
of all Omegas - one constraint

%populate imposition of cluster order
for i=1:I_tot
    for j_alt=1:J(i,1)
        for k_alt=1:K_tot
            for j=1:J(i,1)
                for k=1:K_tot
                    if j_alt<j && k_alt>k
                        [col_posX0,~,~] =
findVariableX(variables,X_tot,'X',i,[],j,[],k,[],[]);
                        A(row_Counter,col_posX0) = 1;

                        [~,~,col_posX2] =
findVariableX(variables,X_tot,'X',i,[],[],j_alt,[],k_alt,[]); %finds valid but
alternate j to above for that same i
                        A(row_Counter,col_posX2) = 1;

                        b(row_Counter,1) = 1;

                        row_Counter = row_Counter + 1;
                    end
                end
            end
        end
    end
end
end
end
end

```

```

%populate only one of each k for X
for k=1:K_tot
    for i=1:I_tot
        for j=1:J(i,1)
            [col_posX0,~,~] = findVariableX(variables,X_tot,'X',i,[],j,[],k,[],[]);
            Aeq(row_eqCounter,col_posX0) = 1;
            beq(row_eqCounter,1) = 1;
        end
    end
    row_eqCounter = row_eqCounter + 1; %placed here because sum of every case for
that k must be created
end

%populate only one of each valid ij combo for X
for i=1:I_tot
    for j=1:J(i,1)
        for k=1:K_tot
            [col_posX0,~,~] = findVariableX(variables,X_tot,'X',i,[],j,[],k,[],[]);
            Aeq(row_eqCounter,col_posX0) = 1;
            beq(row_eqCounter,1) = 1;
        end
        row_eqCounter = row_eqCounter + 1; %placed here because sum of every k for
that valid ij combo must be created
    end
end

%populate lb and ub
lb = zeros(nVars,1);
lb=sparse(lb);

ub = Inf(nVars,1);
ub(1:X_tot,1) = 1; %X variables are binary

%populate f (objective function)
f = zeros(nVars,1);
f=sparse(f);
for i = X_tot+1:X_tot+Phi_tot %for the sum of every Phi
    f(i,1) = 1; %activates Phis
end

end

%% finding parameters functions
function [R] = findVariableR(listR,J_tot,prefix,i_alt,j_alt) %Local function 0.1;
searches for reconfiguration time

R=0;
tempR=[i_alt,j_alt];
for a=1:J_tot
    if listR{a,1}==prefix
        if isequal(listR{a,2},tempR)
            R=a;
        end
    end
end
end

function [T] = findVariableT(listT,J_tot,prefix,i,j) %Local function 0.2; searches
for processing time

T=0;
tempT=[i,j];
for a=1:J_tot
    if listT{a,1}==prefix
        if isequal(listT{a,2},tempT)
            T=a;
        end
    end
end

```

```

    end
end
end

%% finding variables functions
function [col_posX0,col_posX1,col_posX2] =
findVariableX(variables,X_tot,prefix,i,i_alt,j,j_alt,k,k_alt,k_next) %Local
function 2.1; searches for X's

col_posX0=0;
col_posX1=0;
col_posX2=0;

tempX0=[i,j,k];
tempX1=[i_alt,j_alt,k_next];
tempX2=[i,j_alt,k_alt];

for a=1:X_tot
    if variables(a,1)==prefix
        if isequal(variables(a,2),tempX0)
            col_posX0=a;
        else if isequal(variables(a,2),tempX1)
            col_posX1=a;
        else if isequal(variables(a,2),tempX2)
            col_posX2=a;
        end
        end
    end
end
end

function [col_posPhi] =
findVariablePhi(variables,X_tot,nVars,prefix,i,i_alt,j,j_alt,k,k_alt) %Local
function 2.2; searches for Phi

col_posPhi=0;

tempPhi=[i,i_alt,j,j_alt,k,k_alt];

for a=X_tot+1:nVars
    if variables(a,1)==prefix
        if isequal(variables(a,2),tempPhi)
            col_posPhi=a;
        end
    end
end
end

function [col_posOmega] =
findVariableOmega(variables,X_tot,Phi_tot,Omega_tot,prefix,i,i_alt,j,j_alt,k,k_alt)
%Local function 2.3; searches for Omega

col_posOmega=0;

tempOmega=[i,i_alt,j,j_alt,k,k_alt];

for a=X_tot+Phi_tot+1:X_tot+Phi_tot+Omega_tot
    if variables(a,1)==prefix
        if isequal(variables(a,2),tempOmega)
            col_posOmega=a;
        end
    end
end
end

```


end

B.3. Stage III Heuristic

```
%% Input I
tic

%% Initialising
I_ = I'; %transpose of I %rows=parts; columns=fixtures %easier to work with data in
this form hereof
[rows, columns] = size(I_);
fixtures = columns;
parts = nnz(I); %total number of parts

rng(1969, 'twister'); %for repeatable random number generation
R = %randi([30 90],rows,columns); %randomly generated reconfiguration times between
range of 30-90 seconds
T = randi([30 90],rows,columns); %randomly generated part processing times between
range of 30-90 seconds

initial = nan(rows,columns); %matrix of NaN elements only

A = initial; %initialising A
active = find(I_~=0); %identify active elements of I
A(active) = 1; %creates active matrix by overwriting initial A

B = initial;
heads = 1 : rows : rows*columns; %identify first element for each column
pivot = 1; %heads(randperm(numel(heads),1)); %random selection of one of the heads
B(pivot) = 1; %creates initial pivot matrix

C = initial;
candidates = heads(heads~=pivot); %remaining heads
C(candidates) = 1; %creates initial preliminary candidates matrix

AC = A.*C; %valid candidates matrix

jobList = cell(parts,6);
total_idleTime = 0;
total_makespan = R(pivot); %initial reconfiguration time
count = 1;
jobList{count,1} = sprintf('Cell 1');
jobList{count,2} = sprintf('Cell 2');
jobList{count,3} = sprintf('Reconfiguration Time (s)');
jobList{count,4} = sprintf('Part Processing Time (s)');
jobList{count,5} = sprintf('Idle Time (s)');
jobList{count,6} = sprintf('Forced computation used');

%% Heuristic
while AC(~isnan(AC)) %while AC does not comprise entirely of NaNs

    %% Standard computation
    count = count + 1;
    a1 = B.*T; %pivot T value matrix
    a2 = AC.*R; %candidate R values matrix
    a3 = a1(~isnan(a1)); %extract pivot T value only
    a4 = a3-a2; %idle times (T-R) matrix
    a5 = abs(a4); %idle times absolute (|T-R|) matrix
    [a6,a7] = min(a5(:)); %minimum idle time value (a6), best candidate element
number (a7)
    [~, a8] = ind2sub(size(I_), pivot); %column of pivot - indicates which fixture
it belongs to
    [~, a9] = ind2sub(size(I_), a7); %column of best candidate - indicates which
fixture it belongs to
    jobList{count,6} = 0;
    %
    %% Forced computation for lag column
    b1 = sum(A==1); %remainders for each column - a row vector
    [b2,b3] = max(b1); %maximum remainder, column
```

```

    b4 = sum(b1)-b2; %sum of other remainders
    if b2-b4 >= 0 && b3~=a8 %lag column condition - if max remainder is the sum of
other remainders AND if column of max remainder is not the same as the column of
the pivot used
        [b5 b6] = find(AC==1); %candidate row number, candidate column number
        b7 = sub2ind(size(AC), b5(b6==b3), b3); %lag column candidate element
number
        b8 = R(b7); %lag column candidate R time
        b9 = a3-b8; %idle time calculation (T-R)
        b10 = abs(b9); %idle time absolute (|T-R|)
        %overwriting original values
        a6 = b10; %overwriting minimum idle time value
        a7 = b7; %overwriting best candidate element number
        a9 = b3; %overwriting column of best candidate

        sprintf('Forced computation used for time period: %d', count)
        jobList{count,6} = 1;
    end
    %}
    %% Writing values to schedule
    jobList{count,1} = [a9, I_(a7)]; %fixture-part mapping in Cell 1
    jobList{count,2} = [a8, I_(pivot)]; %fixture-part mapping in Cell 2
    jobList{count,3} = R(a7); %reconfiguration time for that time period
    jobList{count,4} = a3; %part processing time for that time period
    jobList{count,5} = a6; %idle time for that time period
    total_idleTime = total_idleTime + a6; %total idle time counter
    maxTime = max([R(a7) a3]); %maximum operation time between Cell 1 and Cell 2
    total_makespan = total_makespan + maxTime; %total makespan counter

    %% Editing matrices for next run
    A(pivot)=nan; %renders previous pivot invalid for future candidacy
    B(pivot)=nan; %reversing previous pivot
    if pivot ~= numel(I_) %if pivot is not the last element of I (would not work if
last element, redundant operation if last element anyway)
        C(pivot+1)=1; %element directly after previous pivot is now candidate for
that fixture; A matrix cancels out candidacy if operation continues to the next
column
    end
    pivot=a7; %new pivot (previous best candidate)
    B(pivot)=1; %forms new pivot matrix
    C(pivot)=nan; %forms new preliminary candidates matrix
    AC = A.*C; %new valid candidates matrix; A overwrites previous pivots in C for
when C(pivot+1)=1 is operated on last element in a column (which writes the 1 to
the next column - a previous pivot and thus invalid candidate)

end

%% Finalising
total_makespan = total_makespan + T(pivot); %adding final part processing time

t_Heuristic = toc

%% AnyLogic data synthesis
count = ones(fixture,1); %to monitor intracluster sequence

AL_fixture_generation = zeros(fixture,5);
levels = 6; %ASRS rows
positions = 7; %ASRS columns
for i=1:fixture
    AL_fixture_generation(i,1) = i; %fixture_name
    AL_fixture_generation(i,2) = 0; %intra_sequence
    AL_fixture_generation(i,3) = i-1; %storage_space
    AL_fixture_generation(i,4) = floor((i-1)/levels); %storage_position
    AL_fixture_generation(i,5) = (i-1) - ((floor((i-1)/levels))*6); %storage_level
end
headerFixture = {'fixture_name', 'intra_sequence', 'storage_space',
'storage_position', 'storage_level'};
xlswrite('AL_fixture_generation',headerFixture,'Sheet1','A1');

```

```

xlswrite('AL_fixture_generation',AL_fixture_generation,'Sheet1','A2');

AL_part_generation = zeros(parts,1);
for i=1:parts
    AL_part_generation(i,1) = i; %part_name
end
headerPart = {'part_name'};
xlswrite('AL_part_generation',headerPart,'Sheet1','A1');
xlswrite('AL_part_generation',AL_part_generation,'Sheet1','A2');

AL_main_data = zeros(parts,6);
for i=1:parts-1
    AL_main_data(i,1) = i; %row_index
    AL_main_data(i,2) = jobList{i+1,2}(1); %fixture_name
    AL_main_data(i,3) = count(jobList{i+1,2}(1)); %intra_sequence
    AL_main_data(i,4) = R(AL_main_data(i,3), AL_main_data(i,2)); %fr_delay (R)
    AL_main_data(i,5) = jobList{i+1,2}(2); %part_name
    AL_main_data(i,6) = T(AL_main_data(i,3), AL_main_data(i,2)); %pp_delay (T)

    count(jobList{i+1,2}(1)) = count(jobList{i+1,2}(1)) + 1; %increase counter for
intra_sequence

    if i+1 == parts %for final part
        AL_main_data(i+1,1) = i+1; %row_index
        AL_main_data(i+1,2) = jobList{i+1,1}(1); %fixture_name (from column 1)
        AL_main_data(i+1,3) = count(jobList{i+1,1}(1)); %intra_sequence (from
column 1)
        AL_main_data(i+1,4) = R(AL_main_data(i+1,3), AL_main_data(i+1,2));
%fr_delay (R)
        AL_main_data(i+1,5) = jobList{i+1,1}(2); %part_name (from column 1)
        AL_main_data(i+1,6) = T(AL_main_data(i+1,3), AL_main_data(i+1,2));
%pp_delay (T)
    end
end
headerMain =
{'row_index','fixture_name','intra_sequence','fr_delay','part_name','pp_delay'};
xlswrite('AL_main_data',headerMain,'Sheet1','A1');
xlswrite('AL_main_data',AL_main_data,'Sheet1','A2');

```

C. Appendix C: Standard Operating Procedures

C.1. Fixture Fabrication

C.1.1. Introduction

Operator to fabricate initial fixtures for Fixture Storage inventory. Fixture quantity as per designated instructions.

C.1.2. Task Procedure

- 1) Ensure surrounding conditions are safe for operation commencement.
- 2) Release CNC router emergency stop button.
- 3) Manually jog CNC router tool tip to maximum height above machine workbed for sufficient clearance.
- 4) Press emergency stop button.
- 5) Retrieve fixture raw material (plain base plate) from Storage Rack for fixture raw materials.
- 6) Clamp plain base plate to CNC router workbed firmly, ensuring excessive movement is not possible.
- 7) Release emergency stop button.
- 8) Manually jog CNC router tool tip to top left corner of plain base plate (or as stated otherwise by operation G-code).
- 9) Manually jog tool tip height to make contact with plain base plate surface.
- 10) Run G-code for fixture fabrication procedure.
- 11) Retrieve 16 pin modules from Storage Rack during G-code operation run.
- 12) Press emergency stop button upon completion of operation, or if erroneous operating behaviour is observed.
- 13) Unclamp fixture base plate from workbed.
- 14) Visually inspect base plate for defects.
- 15) If defects found: Place base plate aside. Log in Notes/Observations. Continue with fabrication of next fixture (Step 1).
- 16) If no defects found: Insert the 16 pin modules on base plate in an arrangement from top left hole until end of second row.
- 17) Dispatch completed fixture to Fixture Storage via conveyor leading to the ASRS.
- 18) Repeat steps until fixture inventory quantity is satisfied.

C.1.3. Notes/Observations

Record fixture number and nature of defects for unsatisfactory base plates:

C.2. Fixture Reconfiguration

C.2.1. Introduction

Operator to reconfigure incoming fixtures to required configuration.

C.2.2. Task Procedure

- 1) Check status of next fixture to be dispatched to Part Processing Cell.
- 2) Enter fixture number (or corresponding RFID tag number of pallet) into GUI to retrieve fixture from ASRS.
- 3) If fixture configuration is required as-is: Ensure Fixture Reconfiguration Mode is off (Switch 0L in leftward position). No further action required for said fixture.
- 4) If fixture is to be reconfigured: Switch Reconfiguration Mode on by moving Switch 0L to rightward position.
- 5) Check reconfiguration requirements and retrieve additional pins from Storage Rack if necessary.
- 6) Retrieve fixture from incoming branch conveyor after actuator diverts pallet along said conveyor.
- 7) Remove all mismatched pins as per required pin configuration.
- 8) Insert all new pins as per required pin configuration.
- 9) Set aside additional pins if required.
- 10) Visually inspect pin configuration against requirements. Make corrections if necessary.
- 11) Dispatch fixture on pallet along outgoing branch conveyor when Part Processing Cell is available.
- 12) Log updated pin configuration status to corresponding fixture name/RFID tag number.

C.2.3. Notes/Observations

Make note of any fixture defects observed during handling of said fixture:

C.3. Part Processing

C.3.1. Introduction

Operator to process parts with CNC engraver on the corresponding fixture.

C.3.2. Task Procedure

- 1) Ensure surrounding conditions are safe for operation commencement.
- 2) Notify Fixture Manufacturing Cell of Part Processing Cell availability.
- 3) Retrieve unfinished part to be processed.
- 4) Receive reconfigured fixture from Fixture Manufacturing Cell.
- 5) Clamp fixture on workbed of CNC engraver.
- 6) Place part on fixture, ensuring transverse movement is within specified tolerance.
- 7) Release CNC engraver emergency stop button.
- 8) Manually jog CNC engraver to datum position, based on G-code instructions for corresponding part.
- 9) Run G-code for that part processing operation.
- 10) Press emergency stop button upon completion of operation, or if erroneous operating behaviour is observed.
- 11) Unclamp fixture from workbed.
- 12) Remove finished part from fixture.
- 13) Dispatch fixture to Fixture Storage via conveyor leading to ASRS, for recirculation.
- 14) Visually inspect part for defects.
- 15) If defects found: Place part aside. Log in Notes/Observations. Continue with processing of next part (Step 1).
- 16) If no defects found: Dispatch finished part to Part Dispatch for packaging.

C.3.3. Notes/Observations

Record part number and nature of defects for unsatisfactory parts:

D. Appendix D: Catalogue Components

D.1. Mean Well® 12 V PSU

Table D.1: Mean Well ® NES-350-12 specifications [117]

OUTPUT	
DC voltage	12V
Rated current	29A
Current range	0 ~ 29A
Rated power	348W
Ripple & noise (max.)	150mVp-p
Voltage adj. Range	10 ~ 13.5V
Voltage tolerance	±1.5%
Line regulation	±0.5%
Load regulation	±1.0%
Setup, rise time	1000ms, 50ms/230VAC
Hold up time (typ.)	20ms/230VAC
INPUT	
Voltage range	180 ~ 264VAC by switch
Frequency range	47 ~ 63Hz
Efficiency	83%
AC current (typ.)	4A/230VAC
Inrush current (typ.)	40A/115VAC 60A/230VAC
Leakage current	<3.5mA / 240VAC
PROTECTION	
Over load	105 ~ 135% rated output power
Over voltage	13.8 ~ 16.2V 5C
Over temperature	90°C±5C(3.3~7.5V); 85°C±5°C(12~15V)
FUNCTION	
Fan on/off control(typ.)	RTH2≥55°C fan on, ≤50°C fan off (12 ~ 48V)
ENVIRONMENT	
Working temp.	-20 ~ +60°C
Working humidity	20 ~ 90% RH non-condensing
Storage temp., humidity	-20 ~ +85°C, 10 ~ 95% RH
Temp. Coefficient	±0.03%/°C (0 ~ 50°C)
Vibration	10 ~ 500Hz, 3G 10min./1cycle, 60min. each along X, Y, Z axes
SAFETY	
Safety standards	UL60950-1 approved
Withstand voltage	I/P-O/P:3KVAC; I/P-FG:2KVAC; O/P-FG:0.5KVAC
Isolation resistance	I/P-O/P, I/P-FG, O/P-FG:100M Ohms/500VDC / 25°C/ 70% RH
OTHERS	
MTBF	234.3K hrs min.; MIL-HDBK-217F (25°C)
Dimension	215*115*50mm (L*W*H)
Packing	1.07Kg; 12pcs/13.5Kg/0.92CUFT

D.2. Huaguan Relays® 24 V DC Relay

Table D.2: NT72-2C-S10 specifications [118]

CONTACT DATA	
Contact Arrangement	1C(SPDT(B-M))
Contact Material	AgCdO AgSnO2
Contact Rating (resistive)	10A
Max. Switching Power	336W 2400VA
Max. Switching Voltage	110VDC 380VAC

Max. Switching Current:	15A
Contact Resistance or Voltage drop	≤50mΩ
Operation life (Electrical)	10 ⁵
Operation life (Mechanical)	10 ⁷
COIL PARAMETERS	
Coil voltage (Rated)	12VDC
Coil voltage (Max.)	15.6VDC
Coil resistance	400Ω±10%
Pickup voltage	9.00VDC (max)
Release voltage	1.2VDC (min)
Coil power consumption	0.36W
Operate Time	< 7ms
Release Time	< 4ms
OPERATION CONDITIONS	
Insulation Resistance	500M min (at 500VDC)
Dielectric Strength (Between contacts)	50Hz 1000V
Dielectric Strength (Between contact and coil)	50Hz 4000V
Shock resistance	100m/s ² ;11ms
Vibration resistance	10~55Hz double amplitude 1.5mm
Terminals strength	10N
Solderability	235°C ±2°C 3±0.5s
Ambient Temperature	-40~85°C
Relative Humidity	85% (at 40°C)
Mass	11g

D.3. Festo® Standard Cylinder

Table D.3: DSBC-32-500-PPSA-N3 specifications [119]

FEATURE	VALUE
Stroke	500 mm
Piston diameter	32 mm
Piston rod thread	M10×1,25
Cushioning	PPS: Self-adjusting pneumatic end-position cushioning
Assembly position	Any
Conforms to standard	ISO 15552
Piston-rod end	Male thread
Design structure	Piston; Piston rod; Profile barrel
Position detection	For proximity sensor
Variants	Single-ended piston rod
Working pressure	0.6 – 12 bar
Mode of operation	double-acting
Operating medium	Compressed air in accordance with ISO8573-1:2010 [7:4:4]
Note on operating and pilot medium	Lubricated operation possible (subsequently required for further operation)
Corrosion resistance classification CRC	2 - Moderate corrosion stress
Ambient temperature	-20 – 80 °C
Impact energy in end positions	0.4 J
Cushioning length	17 mm
Theoretical force at 6 bar, return stroke	415 N
Theoretical force at 6 bar, advance stroke	483 N
Moving mass with 0 mm stroke	110 g
Additional weight per 10 mm stroke	27 g
Basic weight for 0 mm stroke	465 g
Additional mass factor per 10 mm of stroke	9 g
Mounting type	Optional, with internal (female) thread, with accessories
Pneumatic connection	G1/8
Materials note	Conforms to RoHS
Materials information for cover	Aluminium die cast, coated
Materials information for seals	TPE-U(PU)
Materials information for piston rod	High alloy steel
Materials information for cylinder barrel	Wrought Aluminium alloy; Smooth anodized

D.4. Festo® One-Way Flow Control Valve

Table D.4: GRLA-1/8-QS-6-D specifications [120]

FEATURE	VALUE
Valve function	One-way flow control function for exhaust air
Pneumatic connection, port 1	QS-6
Pneumatic connection, port 2	G1/8
Adjusting element	Slotted head screw
Mounting type	Threaded
Standard nominal flow rate in flow control direction	185 l/min
Standard nominal flow rate in non-return direction	160 – 240 l/min
Ambient temperature	-10 – 60 °C
Assembly position	Any
Operating pressure complete temperature range	0.2 – 10 bar
Standard flow rate in direction of flow control: 6 -> 0 bar	370 l/min
Standard flow rate in blocked direction: 6 -> 0 bar	330 – 390 l/min
Operating medium	Compressed air in accordance with ISO8573-1:2010 [7:4:4]
Note on operating and pilot medium	Lubricated operation possible (subsequently required for further operation)
Medium temperature	-10 – 60 °C
Nominal tightening torque	3 Nm
Tolerance for nominal tightening torque	± 10 %
Product weight	22 g
Materials information for screw-in stud	Wrought Aluminium alloy
Materials note	Conforms to RoHS
Materials information for seals	NBR
Release ring material data	POM
Regulating screw material data	Brass
Swivel joint material data	Zinc die-casting; Chromed

D.5. Festo® Solenoid Valve

Table D.5: VUVS-L20-M52-AD-G18-F7-1C1 specifications [121]

FEATURE	VALUE
Valve function	5/2 monostable
Type of actuation	electrical
Valve size	21 mm
Standard nominal flow rate	700 l/min
Working pressure	2.5 – 10 bar
Design structure	Piston slide
Type of reset	Air spring
Protection class	IP65 to IEC 60529 with plug socket
Nominal size	5.7 mm
Exhaust-air function	throttleable
Sealing principle	soft
Assembly position	Any
Manual override	Detenting; Pushing
Type of piloting	Piloted
Pilot air supply	Internal
Flow direction	non reversible
Freedom from overlap	Yes
b value	0.35
C value	2.9 l/sbar
Switching time off	29 ms
Switching time on	20 ms
Duty cycle	100%
Max. positive test pulse with logic 0	1,900 µs
Max. negative test pulse with logic 1	2,700 µs
Characteristic coil data	24 V DC: 2.6 W
Permissible voltage fluctuation	+/- 10 %

Operating medium	Compressed air in accordance with ISO8573-1:2010 [7:4:4]
Note on operating and pilot medium	Lubricated operation possible (subsequently required for further operation)
Vibration resistance	Transport application test at severity level 2 in accordance with FN942017-4 and EN 60068-2-6
Shock resistance	Shock test with severity level 2 in accordance with FN 942017-5 and EN 60068-2-27
Corrosion resistance classification CRC	2 - Moderate corrosion stress
Medium temperature	-10 – 60 °C
Pilot medium	Compressed air in accordance with ISO8573-1:2010 [7:4:4]
Ambient temperature	-10 – 60 °C
Product weight	222 g
Electrical connection	Design C
Mounting type	Optional on manifold rail with through hole
Scavenging orifice connection	Non-ducted
Pilot exhaust port 84	M5
Pneumatic connection, port 1	G1/8
Pneumatic connection, port 2	G1/8
Pneumatic connection, port 3	G1/8
Pneumatic connection, port 4	G1/8
Pneumatic connection, port 5	G1/8
Materials note	Conforms to RoHS
Materials information for seals	HNBR; NBR
Materials information, housing	Aluminium die cast; Painted
Material information, piston spool	Wrought Aluminium alloy
Screw material data	steel, galvanized

D.6. Festo® Silencer

Table D.6: AMTE-M-H-G18 specifications [122]

FEATURE	VALUE
Assembly position	Any
Container size	20
Working pressure	0 – 10 bar
Flow rate to atmosphere	615 l/min
Operating medium	Compressed air in accordance with ISO8573-1:2010 [7:-:-]
Note on operating and pilot medium	Lubricated operation possible
Corrosion resistance classification CRC	1 - Low corrosion stress
Sound pressure level	92 dB(A)
Ambient temperature	-40 – 80 °C
Product weight	5 g
Pneumatic connection	G1/8
Materials information for silencer insert	Bronze
Materials information for screw-in stud	Brass
Materials note	Conforms to RoHS

D.7. Festo® Diffuse Light Sensor

Table D.7: SOEG-RT-M18-PA-K-2L specifications [123]

FEATURE	VALUE
Design	Round
Conforms to standard	EN 60947-5-2
CE symbol (see declaration of conformity)	according to EU-EMV guideline
Materials note	Free of copper and PTFE; Contains PWIS substances
Measured variable	Position
Measuring principle	Optoelectronic
Measurement method	Diffuse reflection sensor
Type of light	Red
Working range	40 – 600 mm

Ambient temperature	-25 – 55 °C
Switch output	PNP
Switching element function	Antivalent
Hysteresis	<= 60 mm
Max. switching frequency	1,000 Hz
Max. output current	200 mA
Voltage drop	2 V
Short circuit strength	Pulsing
Operating voltage range	DC 10 – 36 V
Residual ripple	20 %
Idle current	20 mA
Polarity protected	for all electrical connections
Electrical connection	Cable; 4-core
Cable length	2.5 m
Materials information, cable sheaths	TPE-U(PUR)
Material information, isolating sleeve	PVC
Size	M18
Mounting type	with lock nut
Tightening torque	20 Nm
Assembly position	Any
Product weight	121 g
Materials information, housing	Brass; Chromed-plated
Operating status display	Yellow LED
Operating reserve display	Green LED
Setting options	Potentiometer
Setting range lower limit	40 mm
Upper limit of adjustment range	600 mm
Ambient temperature with flexible cable installation	-5 – 55 °C
Protection class	IP65; IP67
Corrosion resistance classification CRC	2 - Moderate corrosion stress