UNIVERSITÉ DU LUXEMBOURG

PhD-FSTM-2021-013
The Faculty of Sciences, Technology and Medicine

DISSERTATION

Defence held on 04/02/2021 in Luxembourg

to obtain the degree of

DOCTEUR DE L'UNIVERSITÉ DU LUXEMBOURG

EN SCIENCES DE L'INGENIEUR

by

Sebastian Groß
Born on 21 January 1993 in Lebach, (Germany)

AGILE FERTIGUNGSSTEUERUNG FÜR (RE-)FABRIKATIONSSYSTEME

Dissertation defence committee

Prof. Dr.-Ing. Peter Plapper, dissertation supervisor
Professor, Université du Luxembourg

Prof. Dr.-Ing. Wolfgang Gerke, Vice Chairman
Professor, Hochschule Trier – Umwelt-Campus Birkenfeld

Prof. Dr.-Ing. Frank Scholzen, Chairman
Professor, Université du Luxembourg

Prof. Dr.-Ing. Michael Freitag
Professor, Universität Bremen

Prof. Dr.-Ing. Rainer Müller
Professor, Univerität des Saarlandes

Kurzfassung

Steigende Produktvarianz und –individualisierung führen zu steigenden Flexibilitätsansprüchen gegenüber der Fertigung und der Fertigungssteuerung. Zusätzlich verschärfen sich diese Anforderungen im Rahmen der Refabrikation durch unbekannte Zustände der gebrauchten Produkte nochmals. Jedes zu refabrizierende Produkt kann daher eine individuelle Route durch das Refabrikationssystem benötigen. Dieser Prozess, in welchem gebrauchte Produkte in einen neuwertigen Zustand transformiert werden, gewinnt durch hohe ökologische und ökonomische Potenziale sowie gesetzliche Vorschriften zunehmende Aufmerksamkeit. Um entsprechenden Anforderungen gerecht zu werden, erfolgt die Vorstellung einer hybriden Steuerungsarchitektur. Diese besteht aus zentralen und dezentralen Komponenten. In der dezentralen Ebene werden alle physischen Fertigungsteilnehmer mit softwaretechnischen Komponenten vernetzt und von diesen gesteuert. Diese Komponenten können den Status sowie die Verfügbarkeit der zugehörigen Fertigungsteilnehmer akquirieren. Sie können sowohl untereinander als auch mit der zentralen Ebene kommunizieren. In der zentralen Ebene erfolgt die Ablaufplanung von Maschinen und fahrerlosen Transportsystemen (FTS). Diese wird simultan und nicht, wie bei den derzeit verfügbaren Steuerungssystemen sequenziell durchgeführt. Zur Optimierung der Ablaufplanung wird ein auf Constraint Programming (CP) basierendes Verfahren entwickelt. Simulationsergebnisse zeigen, dass eine simultane, gegenüber einer sequenziellen, Ablaufplanung eine Durchlaufzeitreduzierung von 35,6 % ermöglicht. Der CP-basierte Ansatz liefert im Vergleich zu anderen State of the Art-Verfahren, die besten Ergebnisse. Zusätzlich werden diese in einer deutlich kürzeren Rechenzeit generiert. Die Steuerungsarchitektur ist in der Lage, adäquat auf unerwartet auftretende Ereignisse, wie Maschinenausfälle oder neue Aufträge zu reagieren. Sie nutzt hierzu Echtzeit-Feedback vom Shop-Floor. Die Implementierung der Architektur erfolgt als Multi-Agenten-System. Der Ansatz kann durch die erfolgreiche Steuerung einer Modellfabrik in einer realitätsnahen Umgebung validiert werden.

Increasing product variance and individualisation lead to increasing demands for flexibility in production and production control. In the context of remanufacturing, these demands are further intensified by unknown conditions of the used products. Each product to be remanufactured may therefore require an individual route through the remanufacturing system. This process, which puts used products into an "as good as new or better" condition, is receiving increasing attention due to its high ecological and economic potential and legal regulations. In order to meet these requirements, a hybrid control architecture will be presented. This consists of centralised and decentralised components. At the decentralised level, all physical production participants are networked with software components and controlled by these. These components can acquire the status and availability of the corresponding manufacturing participants. They can communicate with each other as well as with the central level. The central level is where the scheduling of machines and automated guided vehicle (AGVs) takes place. This is carried out simultaneously and not sequentially as is the case with the currently available control systems. A method based on Constraint Programming (CP) is being developed to optimise scheduling. Simulation results show that a simultaneous, as opposed to a sequential, scheduling enables a reduction of makespan by 35.6 %. Compared to other state of the art methods, the CP-based approach provides the best results and this in a significantly shorter computing time. The control architecture is able to react adequately to unexpected events such as machine failures or new orders. It uses real-life feedback from the shop floor for this purpose. The architecture is implemented as a multi-agent system. The approach can be validated by successfully controlling a model factory in a realistic environment.

Vorwort

Die vorliegende Dissertation entstand im Rahmen des grenzüberschreitenden Forschungsclusters „Robotix-Academy“, welches durch den *Europäischen Fonds für regionale Entwicklung* (EFRE) im Rahmen des *INTRREG*-Programmes V A Großregion gefördert wurde. In dieser Zeit arbeitete ich an der Hochschule Trier, Umwelt-Campus Birkenfeld innerhalb des Fachbereichs Umweltplanung und Umwelttechnik. Zudem war ich als Ph.D. Student Teil der Research Unit in Engineering Sciences (RUES), Bereich Intelligent Robotics an der Fakultät für Naturwissenschaften, Technologien und Kommunikation (FTCS) der Universität Luxemburg. Hierbei begleiteten mich viele Menschen, die mich während der Erstellung dieser Dissertation unterstützten, und ohne die ein Gelingen dieser Arbeit nicht möglich gewesen wäre.

Ich möchte mich zuerst herzlich bei den Professoren des Prüfungskomitees bedanken. Mein ganz besonderer Dank gilt Herrn Prof. Dr.-Ing. Plapper sowie Herrn Prof. Dr.-Ing. Gerke, die mir zum einen das Vertrauen schenkten, und die Möglichkeit gaben, dieses Promotionsvorhaben durchzuführen und mir zum anderen während dieser Zeit als Betreuer zur Seite standen. Sie haben mir wertvolle strategische und fachliche Kompetenzen vermittelt und somit in hohem Maße zum erfolgreichen Abschluss meiner Dissertation beigetragen. Auch möchte ich Herrn Prof. Dr. Scholzen einen besonderen Dank für das sehr hilfreiche und konstruktive Feedback bzgl. meiner CET-Berichte aussprechen, was ebenfalls zu einer deutlichen Verbesserung der Arbeit beigetragen hat. Zudem möchte ich mich bei Herrn Prof. Dr.-Ing. Michael Freitag und Prof. Dr.-Ing. Rainer Müller für die Prüfung meiner Dissertation bedanken.

Während meiner Zeit am Umwelt-Campus Birkenfeld hatte ich das Vergnügen, mit vielen Kollegen zusammenzuarbeiten, die mich ebenfalls auf meinem Weg zur Promotion unterstützt haben, mir immer mit Rat und Tat zur Seite standen. Hier geht mein Dank an Dr. Jan Jungbluth, welcher mich gerade während meiner Anfangszeit und der Suche nach einem passenden Promotionsthema sehr unterstützt und motiviert hat. Auch möchte ich mich bei Florian Schäfer, Thomas Bartscherer und Lars Schaupeter für die Unterstützung, die konstruktiven Gespräche und eine tolle Zusammenarbeit über Jahre, bedanken. Besonders hervorheben und bedanken möchte ich mich hier auch bei Lukas Vogt. Dieser hat mich während der kompletten Umsetzung begleitet und mich im Rahmen seiner Projektarbeiten, Abschlussarbeiten und Hiwi-Tätigkeiten, tatkräftig unterstützt.

Ich möchte mich auch bei allen Studenten bedanken, die mich im Rahmen ihrer Projekt- und Abschlussarbeiten sowie Hiwi-Tätigkeiten bei der Umsetzung und Verifikation meiner Forschungsergebnisse unterstützt haben.

Auch ein großes Dankschön an alle Projektpartner, mit denen ich innerhalb des Robotix-Academy Projektes zusammenarbeiten durfte.

Abschließend möchte mich bei meiner Familie bedanken, ohne die ich vermutlich nie in der Lage gewesen wäre, diese Promotion zu beginnen und auch durchzustehen. Meiner Lebensgefährtin Carina möchte ich besonders für das Verständnis und die Unterstützung während meiner Promotion sowie für die Motivation bei Rückschlägen und in schweren Zeiten danken. Ich bedanke mich für euer Vertrauen, die Unterstützung sowie den Rückhalt, den ihr mir auf meinem Weg zur Promotion gegeben habt.

Birkenfeld, den 16.12.2020

Inhaltsverzeichnis

Kurzfassung ..... **II**

Vorwort ..... **III**

Abbildungsverzeichnis ..... **VII**

Tabellenverzeichnis ..... **X**

Abkürzungsverzeichnis ..... **XII**

1 Einleitung **1**

1.1 Ausgangslage und Motivation 1

1.2 Forschungsthematik und Forschungsziel 2

1.3 Aufbau der Arbeit 4

2 Theoretische Grundlagen und Stand der Forschung **5**

2.1 Produktionsplanung und -steuerung sowie spezielle Anforderungen in der Refabrikation 5
2.1.1 Zentrale Produktionssteuerung 6
2.1.2 Dezentrale Produktionssteuerung 7
2.1.3 Hybride Produktionssteuerung 8
2.1.4 Stand der Forschung: Dezentrale Ansätze in der Produktionsplanung und –steuerung 9
2.1.5 Anforderungen an die Produktionsplanung und -steuerung in der Refabrikation 10
2.1.6 Spezielle Ansätze zur Produktionsplanung und –steuerung in der Refabrikation 12

2.2 Feinplanung 14
2.2.1 Ablaufplanungsprobleme 14
2.2.2 Das flexible Job-Shop Scheduling Problem 15
2.2.3 Definition eines (Re-)fabrikationssystems als Flexible Job-Shop Scheduling Problem 16
2.2.4 Dynamische Ablaufplanungsprobleme 17
2.2.5 Optimierungsziele 19

2.3 Algorithmen zur Lösung kombinatorischer Optimierungsprobleme 20
2.3.1 Exakte Optimierungsverfahren 21
2.3.2 Konstruktive Verfahren 22
2.3.3 Verbesserungsverfahren 23
2.3.4 Metaheuristische Verfahren 25
2.3.5 Künstliche Intelligenz Verfahren 27

2.4 Intralogistik mit fahrerlosen Transportsystemen 29
2.4.1 Flottenmanagement-Systeme 30
2.4.2 Simultane Ablaufplanung von Maschinen und fahrerlosen Transportsystemen 31
2.4.3 Stand der Forschung zur simultanen Ablaufplanung von Maschinen und FTS 32

2.5 Forschungslücke und forschungsleitende Hypothese 35

3 Konzept einer hybriden Architektur zur agilen Fertigungssteuerung **37**

3.1 Anforderungen 37

3.2 Konzept der hybriden, dezentral-hierarchischen Steuerungsarchitektur 37
3.2.1 Vertikale und horizontale Integration der Steuerungsarchitektur 40
3.2.2 Koordinator Komponente 42
3.2.3 Scheduler Komponente 43
3.2.4 Produkt Komponenten 44
3.2.5 Ressourcen Komponenten 47
3.2.6 Transport Komponenten 48
3.2.7 Experten Komponenten 49

4 Hybride Steuerungsarchitektur für (Re-)fabrikationssysteme 51

4.1 Entwicklung des Algorithmus zur simultanen Ablaufplanung 51
4.1.1 CP-Modell zur SAMF ohne alternative Maschinen (JSSP-Umgebung) 52
4.1.2 CP-Modell zur SAMF mit alternativen Maschinen (FJSSP-Umgebung) 58
4.1.3 Neuplanung beim Auftreten unerwarteter Ereignisse 60
4.2 Simulationsumgebung 64
4.2.1 Benchmarkinstanzen innerhalb einer JSSP-Umgebung 64
4.2.2 Benchmarkinstanzen innerhalb einer FJSSP-Umgebung 66
4.3 Simulationsstudien innerhalb deterministischer Umgebung 67
4.3.1 Vergleich verschiedener Metaheuristiken zur Lösung von CP-Modellen 67
4.3.2 Vergleich simultane, sequenzielle und selbstorganisierte, simultane Ablaufplanung 68
4.3.3 Vergleich der Nutzung von CP zur SAMF gegenüber anderen State-of-the-Art Verfahren 71
4.3.4 Abhängigkeit zwischen DLZ-Reduktion und t/p-Verhältnis 74
4.3.5 Abhängigkeit zwischen DLZ und Anzahl verfügbarer FTS 75
4.3.6 Vergleich unterschiedlicher Zielfunktionen 78
4.4 Reaktion der hybriden Steuerungsarchitektur auf unerwartet auftretende Ereignisse 79
4.4.1 Reaktion auf Ausfall einer Maschine und auf Ausfall eines FTS 81
4.4.2 Reaktion auf neue, zu berücksichtigende Inspektionsergebnisse 83
4.4.3 Reaktion auf neuen, auszuführenden Auftrag 84
4.4.4 Reaktion auf nacheinander eintreffende, unvorhergesehene Ereignisse 85
4.5 Zwischenfazit 87

5 Validierung der Steuerungsarchitektur 89

5.1 Aufbau der smarten Modellfabrik zur Refabrikation 90
5.1.1 Vernetzung der einzelnen Fertigungsteilnehmer 96
5.1.2 Versuchsszenario 97
5.2 Versuchsdurchführung 99
5.3 Zwischenfazit 103

6 Zusammenfassung und Ausblick 104

6.1 Zusammenfassung und Diskussion der Forschungsergebnisse 104
6.2 Ausblick 107

7 Anhang 109
Anhang A - Liste wissenschaftlicher Veröffentlichungen: 109
Anhang B - Beteiligung an Forschungsanträgen: 109
Anhang C - Anderweitige Projektakquise: 110
Anhang D - Betreute studentische Arbeiten: 110
Anhang E - Betreuung von Lehrveranstaltungen: 111
Anhang F - Ökologische und ökonomische Vorteile der Refabrikation: 113
Anhang G - Komplexität kombinatorischer Optimierungsprobleme 116
Anhang H - Mixed Integer Programming Modell für das Job-Shop Scheduling Problem 118
Anhang I - Job-Shop Scheduling Beispiel mit genetischem Algorithmus: 119
Anhang J - IBM ILOG CP Optimizer 122
Anhang K - Übersicht State-of-the-Art bzgl. SAMF in JSSP- und FJSSP-Umgebung: 126
Anhang L – BI zur SAMF in einer JSSP-Umgebung nach Bilge und Ulusoy [158] 128
Anhang M - Aufgabenzuordnung durch das Kontraktnetz-Protokoll in MAS 131
Anhang N – BI zur SAMF in einer FJSSP-Umgebung nach Kumar et al. [167] 132
Anhang O - CP-Modell zur Beschreibung eines JSSP: 135
Anhang P - CP-Modell zur Beschreibung eines FJSSP: 136
Anhang Q – Detaillierte Simulationsergebnisse 138
Anhang R - Leerfahrtenzeitenmatrix der Modellfabrik 144

Literaturverzeichnis 145

Abbildungsverzeichnis

Abbildung 1: Basis: Anwender und Planer von Industrie 4.0 Anwendungen ab 100 Mitarbeitern (n=364) * Maximal drei Antworten möglich. (Nach [7]) 3
Abbildung 2: Regelkreis zur Terminregelung im Rahmen der Produktionssteuerung nach [17]. 5
Abbildung 3: Klassifikation von Steuerungsarchitekturen (nach [19]). 6
Abbildung 4: Automatisierungspyramide nach [24]. 6
Abbildung 5: Grundstruktur Agent nach [34]. 8
Abbildung 6: Prozessablauf einer Refabrikation [54]. 10
Abbildung 7: Unterschied zwischen traditioneller Fertigungslinie und flexiblem RS. 11
Abbildung 8: JSSP mit drei Auftragen und drei Maschinen sowie zugehöriger, gültiger Ablaufplan als Gantt-Diagramm. 15
Abbildung 9: FJSSP mit zwei Aufträgen und drei Maschinen (M1-M3). 16
Abbildung 10: Darstellung eines RS nach [60]. 17
Abbildung 11: Optimierungsalgorithmen zur Ablaufplanung (in Anlehnung an [114]) 21
Abbildung 12: GTA: (1) Auswahl der Maschine mit dem frühestmöglichen Endzeitpunkt (nach [3]) 22
Abbildung 13: GTA: (2) Auswahl einer Operation mit Startzeitpunkt innerhalb des Betrachtungsintervalls aus der Warteschlange von M2 (nach [3]) 23
Abbildung 14: Initiale Lösung und Definition einer Nachbarschaft *N(x)* bei der LNS. 24
Abbildung 15: Vertauschung der Elemente in der Nachbarschaft *N(x)* und sich ergebende Verletzung der Randbedingungen. 25
Abbildung 16: Neuer, gültiger Ablaufplan mit reduzierter DLZ gegenüber der initialen Lösung in Abbildung 14. 25
Abbildung 17: Pseudo-Code zum Ablauf der Tabu-Suche 26
Abbildung 18: Ablaufschema GA 26
Abbildung 19: FTS *MiR 100* der Firma *Mobile Industrial Robots* 29
Abbildung 20: Überblick über die fünf Hauptaufgaben zur Steuerung von FTS nach [147]. 30
Abbildung 21: Sequenzielle Ablaufplanung unter der Verwendung von MES und Flottenmanagement-System. 30
Abbildung 22: Gantt-Diagramm zur Repräsentation eines gemeinsamen Ablaufplans für Maschinen und FTS 32
Abbildung 23: Forschungsleitende Hypothese und Forschungsfragen 36
Abbildung 24: Hybride, dezentral-hierarchische Steuerungsarchitektur. 38
Abbildung 25: Aufgaben und Zuständigkeiten der einzelnen Ebenen der hybriden Steuerungsarchitektur. 40
Abbildung 26: Format zur Datenübertragung von übergeordnetem System an die hybride Steuerungsarchitektur. 41
Abbildung 27: Zustandsübergangsdiagramm Produkt Komponente. 46
Abbildung 28: Ablaufdiagramm Produkt Komponente 47
Abbildung 29: CP Constraint *alternative(x, [y_1, y_2])* 53
Abbildung 30: Intervallfolgevariable *p* 54
Abbildung 31: Wert der Intervallfolgevariable *p* 54
Abbildung 32: CP Constraint *endBeforeStart(x_i, x_j, z_{ij})* 54

Abbildung 33: CP Constraint *noOverlap([x1, …, x4])* 54
Abbildung 34: CP Constraint noOverlap() mit transition distance matrix 55
Abbildung 35: Anwendung Leerfahrtenmatrix für ein FTS 57
Abbildung 36: Chronologischer Ablaufplan zur Durchführung einer Neuplanung. 62
Abbildung 37: Ausschnitt der Daten bzgl. der als Testproblem genutzten Auftragssätze (Job Set‘s) für *t/p* > 0,25 [149]. 64
Abbildung 38: Layoutvarianten nach dem Testproblem von Bilge und Ulusoy [158]. 65
Abbildung 39: Ausschnitt der BI zur SAMF innerhalb einer FJSSP-Umgebung von Kumar et al. [167] 66
Abbildung 40: Vergleich zwischen LNS und GA zur Lösung von CP-Modellen in JSSP- und FJSSP-Umgebung. Der jeweils beste Wert ist durch Fettschrift hervorgehoben. 67
Abbildung 41: Vergleich zwischen simultaner und sequenzieller Ablaufplanung von Maschinen und FTS. 69
Abbildung 42: Ergebnisse des Vergleichs zwischen simultaner, sequenzieller und selbstorganisierter [183] Ablaufplanung von Maschinen und FTS bei *t/p > 0,25*. 69
Abbildung 43: Ergebnisse des Vergleichs zwischen simultaner, sequenzieller und selbstorganisierter [183] Ablaufplanung von Maschinen und FTS bei *t/p < 0,25*. 70
Abbildung 44: Ergebnisse des Vergleichs zwischen simultaner, sequenzieller und selbstorganisierter [168] Ablaufplanung von Maschinen und FTS bei *t/p > 0,25* und alternativen Maschinen (FJSSP). 71
Abbildung 45: Zusammenhang zwischen DLZ-Reduktion durch SAMF und dem t/p Verhältnis für alle Instanzen von Bilge und Ulusoy mit t/p < 0,25. 74
Abbildung 46: Zusammenhang zwischen DLZ-Reduktion durch SAMF und dem t/p Verhältnis für alle Instanzen von Bilge und Ulusoy mit t/p > 0,25 75
Abbildung 47: Zusammensetzung der DLZ in einer realen DLZ-Umgebung nach [186]. 75
Abbildung 48: Durchschnittliche DLZ in Abhängigkeit der Anzahl an verfügbaren FTS mit den BI von Bilge und Ulusuoy. 76
Abbildung 49: Abhängigkeit zwischen der Anzahl an FTS, ab welchen keine Reduzierung der DLZ mehr erreicht wird und dem *t/p*-Verhältnis der jeweiligen BI. 77
Abbildung 50: Abhängigkeit zwischen der Anzahl an FTS, ab welchen keine Reduzierung der DLZ mehr erreicht wird und der Anzahl an Aufträge innerhalb der jeweiligen BI. 77
Abbildung 51: Abhängigkeit zwischen dem *t/p*-Verhältnis und der Anzahl an Aufträge innerhalb eines der 40 Auftragssätze bzgl. der BI von [158]. 78
Abbildung 52: Simultaner Ablaufplan für Auftragssatz 1 und Layout 1 der BI von Kumar et al. 81
Abbildung 53: Neuer Ablaufplan nach Ausfall von M2 nach 15 ZE. 82
Abbildung 54: Neuer Ablaufplan nach Ausfall von AGV2 nach 15 ZE. 83
Abbildung 55: Neuer Ablaufplan nach zusätzlich hinzugekommener Operation M1(10) für Produkt 1 nach 15 ZE 84
Abbildung 56: Neuer Ablaufplan nach dem Eintreffen eines neuen, zu bearbeitenden Auftrags (Auftrag 6) nach 15 ZE. 85
Abbildung 57: Neuer Ablaufplan nach dem Eintritt des zweiten unerwarteten Ereignisses. 86
Abbildung 58: Neuer Ablaufplan nach dem Eintritt des dritten unerwarteten Ereignisses. 87
Abbildung 59: Vernetzte Fertigungsteilnehmer zu einem CPS. 90
Abbildung 60: Layout der Modellfabrik 91
Abbildung 61: FTS *MiR100* der Firma Mobile Industrial Robots 92
Abbildung 62: 2D-Karte der Modellfabrik zur Navigation und Lokalisierung der MiR100. 92

Abbildung 63: Alternative Demontagestationen 1 und 2 (Vordergrund: Universal Robots UR3; Hintergrund: KUKA iiwa) mit Transportbox (grün) und RFID-Reader (rote Box) zur eindeutigen Identifikation des vorliegenden Produktes.....93
Abbildung 64: Produktmodell(-graph) für die verwendete Kühlmittelpumpe. 93
Abbildung 65: Anlieferung eines Produktes an Demontagestation 1.....94
Abbildung 66: Graphische Benutzeroberfläche des Fertigungssteurers für die Modellfabrik. 96
Abbildung 67: Vernetzung der Fertigungsteilnehmer sowie der Komponenten der hybriden Steuerungsarchitektur untereinander in der Modellfabrik. Die Komponenten kommunizieren ebenfalls über MQTT. 97
Abbildung 68: Bauteile der Zusatzkühlmittelpumpe. 97
Abbildung 69: Initialer Prozessplan aller Produkt innerhalb der Modellfabrik 99
Abbildung 70: Initialer Ablaufplan zur Bearbeitung von drei Produkten innerhalb der Modellfabrik 100
Abbildung 71: Warteliste der MiR100 zur Durchführung einer Transportfahrt.....100
Abbildung 72: Neuer Ablaufplan inkl. des zusätzlich hinzugekommenen Produktes 4 (erste Leerfahrt zum Lager ist nicht grafisch dargestellt, aber wie zu sehen in der Verfügbarkeit des FTS berücksichtigt) 101
Abbildung 73: GUI zur Übermittlung der Inspektionsergebnisse an die Experten Komponente der Inspektionsstation. 102
Abbildung 74: Neuer, gültiger Ablaufplan mit zusätzliche Reinigungsoperation für Produkt 2. 102
Abbildung 75: Kumulierte Neuzulassungen von Januar 2010 bis Ende März 2019, Quelle: [200], [201] 113
Abbildung 76: Ökologische Vorteile am Beispiel der Refabrikation von Startern der Robert Bosch GmbH [202]. 114
Abbildung 77: Initiale Population mit drei Chromosomen C_1, C_2 und C_3. 119
Abbildung 78: GA-Beispiel nach dem Prozessschritt Mutation. 121
Abbildung 79: Neues Chromosom C6 nach dem Prozessschritt der Mutation 121
Abbildung 80: CP Optimizer Ansatz nach [179].....122
Abbildung 81: Lösungsschritte innerhalb von CP Optimizer. 122
Abbildung 82: Beispielhafter Implikationsgraph zur graphischen Darstellung eines 2-SAT Problems. 124
Abbildung 83: Widerspruch innerhalb eines Implikationsgraphen. 124
Abbildung 84: Mögliche Lösungen des beispielhaften, als Implikationsgraphen dargestellten, 2-SAT Problems. 124
Abbildung 85: Daten bzgl. der als Testproblem genutzten Auftragssätze (Job Sets) für *t/p*-Verhältnis > 0,25 [158].....128
Abbildung 86: Transportzeitenmatrix des Layout 1 der Instanzen mit einem *t/p*-Verhältnis < 0,25 [158] in Form einer .data-Datei. 130
Abbildung 87: Zuordnung einer Aufgabe innerhalb des Kontraktnetz-Protokolls. 131
Abbildung 88: Erweiterte BI zur SAMF im Kontext FJSSP von Kumar et al. [167] Teil (1). 132
Abbildung 89: Erweiterte BI zur SAMF im Kontext FJSSP von Kumar et al. [167] Teil (2). 133
Abbildung 90: Erweiterte BI zur SAMF im Kontext FJSSP von Kumar et al. [167] Teil (3). 134

Tabellenverzeichnis

Tabelle 1: Merkmale der Koordinator Komponente 42
Tabelle 2: Merkmale der Scheduler Komponente 43
Tabelle 3: Merkmale der Produkt Komponenten 45
Tabelle 4: Merkmale der Ressourcen Komponenten. 48
Tabelle 5: Merkmale der Transport Komponenten. 49
Tabelle 6: Merkmale der Experten Komponenten. 50
Tabelle 7: Bewertung und Vergleich der verschiedenen Optimierungsalgorithmen für kombinatorische Probleme. 52
Tabelle 8: Exemplarische Leerfahrtmatrix. Die aufgezeigten Fahrzeiten entsprechen den Fahrzeiten aus Layout 1 der BI von Bilge und Ulusoy in Abbildung 38. 57
Tabelle 9: Transportzeitenmatrix für das Testproblem nach Bilge und Ulusoy für *t/p* > 0,25 [158]. 65
Tabelle 10: Vergleich der Ergebnisse verschiedener Optimierungsverfahren für die SAMF (*t/p > 0,25*). 72
Tabelle 11: Vergleich der Ergebnisse verschiedener Optimierungsverfahren für die SAMF *(t/p < 0,25).* 73
Tabelle 12: Vergleich der Ergebnisse verschiedener Optimierungsverfahren für die SAMF mit alternativen Maschinen. 73
Tabelle 13: Vergleich unterschiedlicher ein- und multikriterieller Zielfunktionen für die simultane Ablaufplanung basierend auf den BI von Kumar et al. 79
Tabelle 14: Transportzeitenmatrix zwischen den einzelnen Stationen innerhalb der Modellfabrik inkl. der Auf- und Abladevorgänge der Transportboxen. 95
Tabelle 15: Prozesszeiten der einzelnen Operationen zur exemplarischen Refabrikation der Zusatzkühlmittelpumpen in der Modellfabrik. 98
Tabelle 16: Vorteile der hybriden Steuerungsarchitektur 106
Tabelle 17: Jährliche Einsparung von Rohstoffen und CO_2 Emissionen durch Refabrikation in Europa [203]. HDOR - Heavy-Duty and Off-Road Vehicles; EEE - Electrical and Electronic Equipment 115
Tabelle 18: Kennzahlen des Marktes für Refabrikation in Europa [203] 115
Tabelle 19: Zeitkomplexitäten nach Korte und Vygen [208] (S. 7) 116
Tabelle 20: Übersicht bzgl. State-of-the-Art zur SAMF. 127
Tabelle 21: Transportzeitenmatrix für das Testproblem nach Bilge und Ulusoy für *t/p*-Verhältnis > 0,25 [158]. 129
Tabelle 22: Transportzeitenmatrix für das Testproblem nach Bilge und Ulusoy für *t/p*-Verhältnis < 0,25 [158]. 130
Tabelle 23: Vergleich zwischen LNS als einzellösungs- und GA als mehrpopulationsbasierte Metaheuristik zur Lösung von CP-Modellen anhand der simultanen Ablaufplanung von Maschinen und FTS. 138
Tabelle 24: Vergleich der verschiedenen generellen Verfahren zur Ablaufplanung von Maschinen und FTS anhand der BI aus [149] für: Links: t/p > 0,25 und Rechts: t/p < 0,25. 139
Tabelle 25: Detaillierte Simulationsergebnisse für alle 40 BI von Kumar et al. [167] bzgl. dem Vergleich zwischen simultaner, sequenzieller und selbstorganisierter [168] Ablaufplanung. 140
Tabelle 26: Vergleich der Ergebnisse verschiedener Optimierungsverfahren für die simultane Ablaufplanung von Maschinen und FTS (*t/p > 0,25*). 141

Tabelle 27: Vergleich der Ergebnisse verschiedener Optimierungsverfahren für die simultane Ablaufplanung von Maschinen und FTS *(t/p < 0,25).* 142
Tabelle 28: Vergleich der Ergebnisse verschiedener Optimierungsverfahren für die simultane Ablaufplanung von Maschinen und FTS mit alternativen Maschinen. 143
Tabelle 29: Leerfahrtenzeitenmatrix zwischen den einzelnen Stationen innerhalb der Modellfabrik mit Auf- und Abladevorgang zur Modellierung der Leerfahrten. 144
Tabelle 30: Transportzeitenmatrix zwischen den einzelnen Stationen innerhalb der Modellfabrik ohne Auf- und Abladevorgang. 144

Abkürzungsverzeichnis

ACO	-	Ant Colony Optimization
AGV	-	Automated Guided Vehicle
APS	-	Advanced Planning and Scheduling
APRA	-	Automotive Parts Remanufacturing Association
AS	-	Ant System
AV	-	Anlagenverfügbarkeit
BDI	-	Belief Desire Intention (Glaube Wunsch Absicht)
BEV	-	Battery Electric Vehicle
BI	-	Benchmarkinstanzen
CAD	-	Computer Aided Design
CIM	-	Computer-Integrated Manufacturing
CNC	-	Computerized Numerical Control
CP	-	Constraint Programmierung
CPS	-	Cyber-physisches System
CRM	-	Customer-Relationship-Management
DLZ	-	Durchlaufzeit
EDD	-	Earliest Due Date
EEE	-	Electrical and Electronic Equipment
EFRE	-	Europäischer Fonds für regionale Entwicklung
ERP	-	Enterprise-Resource-Planning
EU	-	Europäische Union
FC	-	Brennstoffzellenfahrzeuge
FCS	-	Finite Capacity Scheduling
FFS	-	Flexibles Fertigungssystem
FJSSP	-	Flexible Job-Shop Scheduling Problem
FMS	-	Flexible Manufacturing System

FTF	-	Fahrerloses Transportfahrzeug
FTS	-	Fahrerloses Transportsystem
GA	-	Genetische Algorithmen
GAE	-	Gesamtanlageneffektivität
GTA	-	Giffler-Thompson Algorithmus
HDOR	-	Heavy-Duty and Off-Road
HFS	-	Holonisches Fertigungssystem
HMS	-	Holonic Manufacturing System
IE	-	Informationsverarbeitende Einheit
IoT	-	Internet of Things
IGBT	-	Insulated-Gate Bipolar Transistor
JSSP	-	Job-Shop Scheduling Problem
KPI	-	Key Performance Indicator
LG	-	Leistungsgrad
LNS	-	Large Neighborhood Search
LPT	-	longest process time
M2M	-	Machine-to-Machine
MA	-	Maschinenauslastung
MAS	-	Multi-Agenten System
MES	-	Manufacturing Execution System
MQTT	-	Message Queuing Telemetry Transport
MKO	-	Multikriterielle Optimierung
MMS	-	Mensch-Maschine-Schnittstelle
MRI	-	Mensch-Roboter-Interaktion
MRK	-	Mensch-Roboter-Kollaboration
MRP II	-	Manufacturing Resource Planning II
MTSP	-	Multiple-Travelling-Salesman-Problem
OEE	-	Overall Equipment Effectiveness

PC	-	Personal Computer
PHEV	-	Plug-In Hybride
PK	-	Pheromonenkonzentration
PPS	-	Produktionsplanung und –steuerung
QR	-	Qualitätsrate
RC	-	Resistor Capacitor
RFID	-	Radio-Frequency Identification
RS	-	(Re-)fabrikationssystem
RT	-	Remaining Time
SAMF	-	Simultane Ablaufplanung von Maschinen und fahrerlosen Transportsystemen
SCADA	-	Supervisory Control and Data Acquisition
SCM	-	Supply-Chain Management
SotA	-	State-of-the-Art
SPT	-	Shortest Process Time
TQM	-	Total-Quality-Management
TSP	-	Travelling-Salesman-Problem
VDA	-	Verband der Automobilindustrie e. V.
VDI	-	Verein Deutscher Ingenieure
VDMA	-	Verband Deutscher Maschinen- und Anlagenbau
WINQ	-	Work In Next Queue
ZE	-	Zeiteinheit

1 Einleitung

Dieses Kapitel gibt eine Einführung in die Forschungsthematik der vorliegenden Dissertation und präsentiert das Forschungsziel. Darüber hinaus wird die Ausgangslage und Motivation zur Durchführung der Arbeit erläutert.

1.1 Ausgangslage und Motivation

Steigende Produktvarianz sowie Produktindividualisierung durch den Kunden, sind ein deutlich beobachtbarer Markttrend. Dies führt zu einem wachsenden Anspruch an die Flexibilität der Fertigung sowie der damit verbundenen Produktionsplanung und –steuerung (PPS). Zentrale PPS werden diesem Anspruch nicht gerecht [1], sind jedoch aktueller Standard in der Industrie. Zentrale Steuerungssysteme beschränken zudem die Möglichkeiten, die sich durch vernetzte Fertigungssysteme im Kontext des Konzeptes Industrie 4.0 ergeben [2]. Eine Vielzahl an Forschern beschäftigt sich aus diesem Grund mit der Entwicklung dezentraler Steuerungsarchitekturen, welche die Selbstorganisation der Fertigung zum Ziel haben. Die einzelnen Teilnehmer der Fertigung, wie Maschinen, Produkte und Aufträge organisieren sich hierbei selbstständig, indem sie miteinander kommunizieren und verhandeln. Dies erlaubt eine hohe Flexibilität sowie eine gute Reaktionsfähigkeit auf unerwartet eintretende Ereignisse. Maschinenausfälle, Engpässe oder neu eintreffende Aufträge sind Beispiele hierfür. Ein generelles Problem bei solchen dezentralen Ansätzen ist, dass durch den Wegfall einer zentralen Einheit, kein globales Optimum erreicht werden kann. Begründet ist dies dadurch, dass keiner der einzelnen Fertigungsteilnehmer einen Gesamtüberblick über das Fertigungssystem besitzt. Mit einer zentralen Einheit, welche einen Überblick über das gesamte Fertigungssystem besitzt, ist eine globale Optimierung hingegen grundsätzlich möglich. Ein weiteres Problem stellt der Umstand dar, dass der Fertigungsplaner durch die Selbststeuerung des Systems schwer dessen Handeln nachverfolgen und –vollziehen kann. Dies mindert die Akzeptanz solcher Ansätze in der Industrie und verhindert deren Einsatz oftmals [3].

Steigende Produktvarianz und –individualisierung stellen ebenfalls neue Ansprüche an die Flexibilität des Materialtransports innerhalb eines Fertigungssystems. Die Fertigungsrouten variieren zwischen den einzelnen Produktvarianten, da nicht jede Variante die gleichen Prozessschritte benötigt, und dies auch nicht in der selben Reihenfolge durchläuft. Dadurch wird ein flexibler Materialtransport notwendig. Eine starre Verkettung von Fertigungsressourcen, bspw. durch Förderbänder, wird solchen Ansprüchen nicht gerecht. Ein vielversprechender Ansatz zur Realisierung eines flexiblen Materialtransports ist der Einsatz fahrerloser Transportsysteme (FTS). Diese erfassen ihre Umgebung durch entsprechende Sensorik und können sich im Raum zweidimensional frei bewegen. Deren steuerungstechnische Integration in die Fertigung erfolgt durch sogenannte Flottenmanagement-Systeme [4]. Diese Systeme erhalten die einzelnen Transportaufträge aus dem Fertigungssteuerungssystem, und ordnen die verfügbaren FTS den einzelnen Transportaufträgen zu. Bei diesem Vorgehen werden zwei, voneinander abhängige und sich gegenseitig beeinflussende, Ablaufplanungsprobleme separat und sequenziell betrachtet. Das erste der beiden, die Maschinenbelegungsplanung, ordnet die durchzuführenden Operationen der einzelnen Aufträge den verfügbaren Maschinen zu. Dieser Schritt geschieht im Rahmen der Feinplanung. Das Flottenmanagement-System ordnet, basierend auf dem zuvor erhaltenen Maschinenbelegungsplan, die vorhanden FTS den sich ergebenden Transportaufträgen zu. Diese beiden Ablaufplanungsprobleme sind einzeln für sich Gegenstand vieler Forschungsarbeiten. Allerdings

gibt es kaum Literatur über die gemeinsame, simultane Betrachtung dieser beiden Probleme. In den wenigen vorhandenen Arbeiten zur simultanen Ablaufplanung von Maschinen und FTS (SAMF) werden fast ausschließlich deterministische Annahmen getroffen, und keine Reaktion der Ansätze auf unerwartet eintretende Ereignisse untersucht. Auch eine Betrachtung alternativer oder paralleler Fertigungsressourcen findet selten statt, ist jedoch von hoher, praktischer Relevanz.

In der Domäne Refabrikation sind die Anforderungen an die Flexibilität der Fertigungssteuerung nochmals erhöht. Die Refabrikation ist ein Prozess, bei welchem gebrauchte Produkte in ihren ursprünglichen Neuzustand versetzt werden. Dieser Prozess geschieht in Übereinstimmung mit spezifischen, technischen Spezifikationen, einschließlich Konstruktions-, Qualitäts- und Teststandards. Er erzeugt Produkte mit voller Garantie, welche einen weiteren Lebenszyklus durchlaufen können [5]. Der erhöhte Flexibilitätsanspruch ergibt sich dadurch, dass die zu refabrizierenden Produkte unterschiedliche Beschädigungen aufweisen können und somit individuelle Refabrikationsprozesse notwendig sind. Selbst für die gleiche Produktvariante können sich unterschiedliche Routen durch das Refabrikationssystem ergeben. Weiterhin ist zu berücksichtigen, dass nach einer Eingangskontrolle des gebrauchten Produktes nicht der komplette Produktzustand bekannt ist. Daher finden während der Prozessausführung weitere Inspektionsvorgänge statt, von deren Ergebnis der weitere Prozessplan abhängig ist. Dies kann dazu führen, dass das Produkt eine komplett neue Route durch das Refabrikationssystem nehmen muss. Das Fertigungssteuerungssystem muss in diesem Fall den neuen Prozessplan in den laufenden Betrieb integrieren, und hierfür den gesamten Ablaufplan an die Änderungen entsprechend anpassen. Spezielle Fertigungssteuerungssysteme für die Refabrikation sind aktuell kommerziell nicht erhältlich. Traditionelle Fertigungssteuerungssysteme werden dem Flexibilitätsanspruch der Refabrikation nicht gerecht. Eine spezielle Lösung zur Fertigungssteuerung von Refabrikationssystemen ist deshalb notwendig.

1.2 Forschungsthematik und Forschungsziel

Die Forschungsthematik der vorliegenden Arbeit lässt sich in den Bereichen PPS, Industrie 4.0 sowie FTS einordnen. In der klassischen PPS hält zunehmend innovative Informations- und Kommunikationstechnik (IKT) Einzug. Mithilfe dieser Technologien sollen Maschinen, Roboter, Menschen, logistische Einheiten sowie Produkte miteinander kommunizieren und kooperieren. Eine umfassende Digitalisierung der industriellen Produktion soll dadurch erreicht werden. Dieses Vorhaben wird von der Forschungsunion der deutschen Bundesregierung unter dem Begriff *Industrie 4.0* zusammengefasst [6]. Zwei dieser Ziele, die durch den Einsatz von Industrie 4.0 Technologien erreicht werden sollen, sind [7] (siehe Abbildung 1):

- Erhöhung der Flexibilität der Produktion
  → schnellere Umsetzung von individuellen Kundenwünschen
- Erhöhung der Kapazitätsauslastung der einzelnen Fertigungsressourcen

Zur Erreichung einer erhöhten Kapazitätsauslastung ist eine Optimierung der Fertigungsfeinplanung notwendig. In dieser wird die Maschinenbelegung und die zugehörige Ablaufplanung definiert. Der aktuelle Status, die Verfügbarkeit sowie die Auslastung aller relevanten Fertigungsressourcen soll bei der Fertigungsfeinplanung Berücksichtigung finden, um einen möglichst effizienten Ablaufplan er-

stellen zu können. Die hierzu notwendigen Daten können durch den Einsatz von Industrie 4.0 Technologien wie dem Internet der Dinge (IoT - Internet of Things) und cyber-physischen Systemen (CPS) erfasst und an das entsprechende Feinplanungstool übermittelt werden.

Für die Realisierung einer schnellen Umsetzung von individuellen Kundenwünschen ist eine flexible und wandlungsfähige Fertigung notwendig. Hierfür muss auch die Intralogistik, bspw. durch den Einsatz von FTS, flexibel gestaltet werden. Diese müssen entsprechend in die Fertigung integriert sowie in der Fertigungsfeinplanung berücksichtigt werden. Die Ablaufplanung bzgl. der Maschinenbelegung und der FTS sind voneinander abhängig und beeinflussen sich gegenseitig. Sie sollten deshalb gemeinsam betrachtet werden, um das Ziel der verbesserten Kapazitätsauslastung bestmöglich erreichen zu können. Die Fertigungsfeinplanung stellt ein Glied innerhalb der PPS dar und ist in die Fertigungssteuerung einzuordnen.

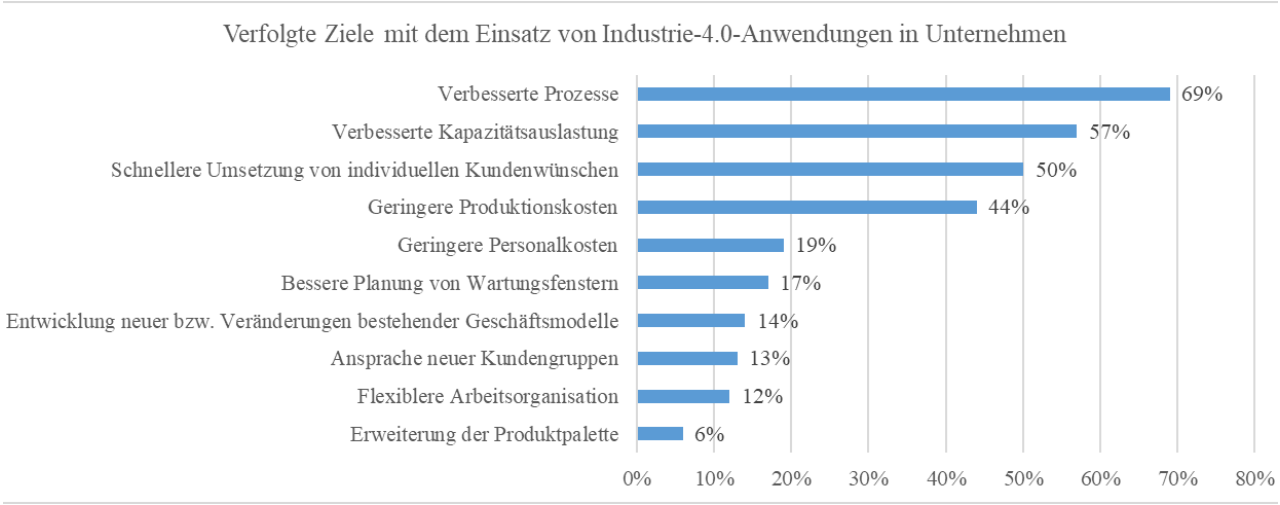


Abbildung 1: Basis: Anwender und Planer von Industrie 4.0 Anwendungen ab 100 Mitarbeitern (n=364) * Maximal drei Antworten möglich. (Nach [7])

Wie eingangs erwähnt, kann bei der Fertigungssteuerung grundsätzlich zwischen zentraler und dezentraler Fertigungssteuerung unterschieden werden. Der Vorteil eines zentralen Ansatzes ist, dass alle globalen Informationen über das Fertigungssystem in die Erstellung des Ablaufplans mit einbezogen werden können. Dies ist eine Grundvoraussetzung zum Erreichen eines globalen Optimums [2], [8]. Nachteilig bei zentralen Ansätzen ist hingegen, dass diese inflexibel und somit wenig geeignet für nicht deterministische Systeme sind [9]. Konträr dazu bieten dezentrale Fertigungssteuerungsansätze eine hohe Flexibilität. Grund hierfür ist, dass sich bei diesem Ansatz die Fertigungsressourcen untereinander selbst organisieren, und somit auf unerwartet auftretende Ereignisse schnell dezentral reagiert werden kann [10][11]. Die Produktionsressourcen passen sich hierfür an die neue Situation selbstständig an und koordinieren diese Änderungen untereinander. Der Wegfall einer zentralen Einheit führt allerdings dazu, dass das Erreichen einer globalen Optimierung unwahrscheinlich ist, da jede einzelne Ressource innerhalb des Systems versucht, ihre eigenen Ziele zu maximieren [12]. Hybride Fertigungssteuerungskonzepte nutzen sowohl Elemente und Funktionalitäten zentraler, als auch dezentraler Ansätze. Sie stellen dadurch einen vielversprechenden Ansatz zur Kombination der jeweiligen Vorteile dar.

Ziel der vorliegenden Arbeit ist die Entwicklung und Untersuchung eines Fertigungssteuerungskonzeptes, das allen in Kapitel 1.1 und 1.2 aufgeführten, steuerungstechnischen Ansprüchen gerecht wird. Dieses muss sowohl einen flexiblen Materialtransport berücksichtigen, als auch in der Lage sein

schnell und adäquat auf unerwartet auftretende Ereignisse zu reagieren. Die hierfür notwendige Reaktionsfähigkeit kann als Agilität bezeichnet werden. Der Begriff *Agilität* kann wie folgt definiert werden: „*Agilität ist die Gewandtheit, Wendigkeit oder Beweglichkeit von Organisationen und Personen bzw. in Strukturen und Prozessen. Man reagiert flexibel auf unvorhergesehene Ereignisse und neue Anforderungen.“* [13] Zusätzlich muss die Fertigungssteuerung eine möglichst gute Optimierungsqualität bzgl. verschiedener KPI wie bspw. Durchlaufzeit (DLZ) und Maschinenauslastung (MA) liefern können, um die Gesamteffektivität des (Re-)fabrikationssystems (RS) sicherzustellen. Aktuelle Entwicklungen aus den Bereichen der PPS, Industrie 4.0 sowie FTS werden hierbei berücksichtigt.

1.3 Aufbau der Arbeit

Ziel der vorliegenden Dissertation ist es, sowohl die methodische Vorgehensweise als auch die Resultate der wissenschaftlichen Arbeit zusammenzufassen. Das Konzept einer neuartigen Steuerungsarchitektur sowie eine mögliche Umsetzung dieser werden beschrieben, bevor abschließend ein Ausblick auf mögliche, weiterführende Forschungsthemen gegeben wird. Die Dissertation gliedert sich wie folgt:

Dieses einleitende Kapitel dient der Vorstellung der aktuellen Problemstellung sowie der Erläuterung der Ausgangslage. Zudem erfolgt eine Einordnung und Definition der Forschungsthematik. Im Rahmen des zweiten Kapitels erfolgt eine Einführung in die für das Verständnis notwendigen Grundlagen sowie eine Darstellung des aktuellen Standes der Technik. Die Identifikation der bestehenden Defizite bzgl. der zuvor herausgearbeiteten Herausforderungen im Bereich der Fertigungssteuerung von RS erfolgt zudem. Anschließend wird in Kapitel 3 das Konzept der hybriden Steuerungsarchitektur zur agilen Fertigungssteuerung in RS präsentiert. Kapitel 4 beschreibt danach die Entwicklung der Steuerungsarchitektur sowie die Validierung dieser anhand von Simulationsstudien und bildet damit das wissenschaftliche Kernkapitel dieser Dissertation. In Kapitel 5 wird die Validierung der vorgestellten Steuerungsarchitektur bzgl. der Steuerung einer am Umwelt-Campus Birkenfeld errichteten Modellfabrik zur Refabrikation durchgeführt. Neben der Zusammenfassung erfolgt im letzten Kapitel auch ein Ausblick über mögliche weiterführende Forschungsaktivitäten.

2 Theoretische Grundlagen und Stand der Forschung

Im Rahmen dieses Kapitels erfolgt die Erläuterung der zum Verständnis dieser Arbeit notwendigen theoretischen Grundlagen. Der zugehörige Stand der Forschung wird ergänzend dargestellt. Die sich ergebende Forschungslücke wird definiert und die Relevanz der vorliegenden Arbeit begründet. Anschließend wird die Forschungsidee zur Schließung dieser Lücke vorgestellt. Die daraus resultierende Forschungshypothese wird abschließend präsentiert und die geplante wissenschaftliche Vorgehensweise zur Validierung erläutert.

2.1 Produktionsplanung und -steuerung sowie spezielle Anforderungen in der Refabrikation

Die Produktionsplanung definiert die Prozesse und Operationen, welche zur Herstellung eines bestimmten Produktes notwendig sind. Die technologischen Randbedingungen zwischen den einzelnen Operationen, wie bspw. Reihenfolgebedingungen, werden hierbei festgelegt und Ressourcen- und Materialbedarfe abgeleitet [14].

Die anschließende Produktionssteuerung, auch Fertigungssteuerung genannt, legt basierend auf den Ergebnissen der Produktionsplanung, die zeitliche Zuordnung der einzelnen Operationen zu den vorhandenen Fertigungsressourcen fest [15] [16]. Auftragsfreigabe und Kapazitätssteuerung sind weitere Aufgaben der Produktionssteuerung [14]. Ein Regelkreis zur Terminierung von Fertigungsaufträgen im Rahmen der Produktionssteuerung wird in [17] vorgestellt und ist in Abbildung 2 zu sehen. In heutigen zentralen Steuerungskonzepten ist dieser Regelkreis immer noch in leicht abgewandelter Form vorzufinden [14].

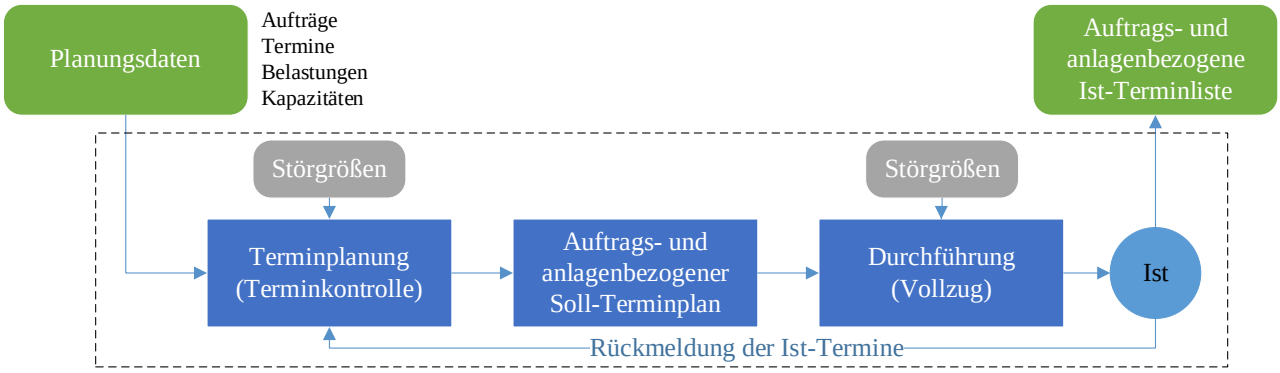


Abbildung 2: Regelkreis zur Terminregelung im Rahmen der Produktionssteuerung nach [17].

Die verschiedenen Steuerungsarchitekturen lassen sich in drei Konzepte unterscheiden (Abbildung 3):

- zentral-hierarchisch (zentral)
- dezentral-hierarchisch (hybrid)
- dezentral-heterarchisch (dezentral)

In produzierenden Betrieben sind überwiegend zentrale Steuerungssysteme vorzufinden [14], [18]. Die verschiedenen Steuerungsarchitekturen werden in den nachfolgenden Kapiteln 2.1.1 bis 2.1.3 erläutert.

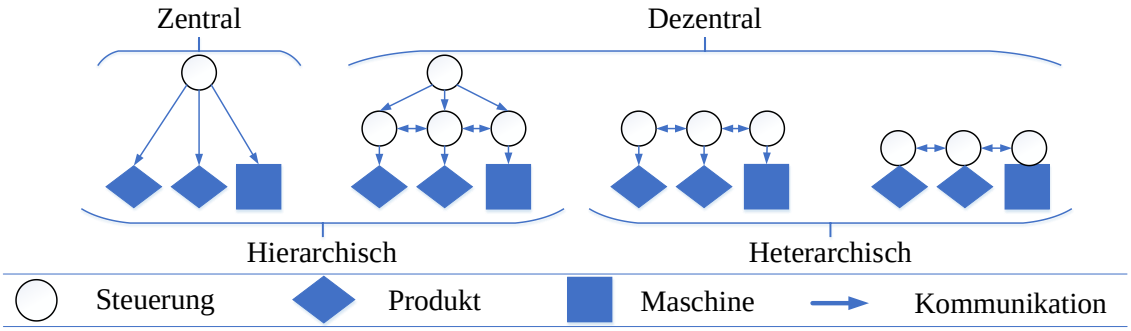


Abbildung 3: Klassifikation von Steuerungsarchitekturen (nach [19]).

2.1.1 Zentrale Produktionssteuerung

Erfolgt die Steuerung der Produktion anhand eines zuvor zentral festgelegten Produktionsplans spricht man von zentral-hierarchischen (folgend als *zentral* bezeichnet) Steuerungsarchitekturen [20]. Dieser Ansatz bietet eine hohe Planungsgenauigkeit und -sicherheit, vorausgesetzt die Vorhersagequalität ist hoch bzw. die Auftragsschwankung gering [21], [22]. Die Vor- und Nachteile zentraler Steuerungsarchitekturen wurden bereits an vorheriger Stelle erläutert. Die aufgeführte, geringe Flexibilität dieser wird im Folgenden detaillierter erläutert. Das Eintreten eines unerwarteten Ereignisses, wie bspw. eine Maschinenstörung, löst bei zentralen Ansätzen eine Neuplanung (Rescheduling) aus. Der aktuelle Status sowie die Verfügbarkeit aller Fertigungsteilnehmer wird mitberücksichtigt und ein neuer, aktualisierter Ablaufplan erstellt. Dieser wird anschließend entweder in den bisher bestehenden Ablaufplan integriert oder ersetzt diesen vollständig. Treten unerwartet Ereignisse in zu kurzen Abständen auf, kann dies bei zentralen Ansätzen durch häufige Neuplanung zur Instabilität des Fertigungssystems führen [23]. Hinzukommt, dass die Daten, auf welchen die Neuplanung basiert, nicht mehr aktuell sind, bis diese alle Ebenen der Hierarchie der zentralen Architektur durchlaufen haben [12]. Aus diesen Gründen sind Ansätze der zentralen Produktionssteuerung inflexibel und wenig geeignet für Systeme, in denen häufig unerwartet Ereignisse eintreten [9]. Klassische *Enterprise-Resource-Planning* (ERP) Systeme und *Manufacturing-Execution-Systems* (MES) sind zwei Vertreter von zentralen Steuerungssystemen und stellen in der Automatisierungspyramide (Abbildung 4) die beiden obersten Ebenen dar. ERP-Systeme repräsentieren hierbei die oberste Ebene (Unternehmensebene) und MES die zweithöchste Ebene (Betriebsleitebene).

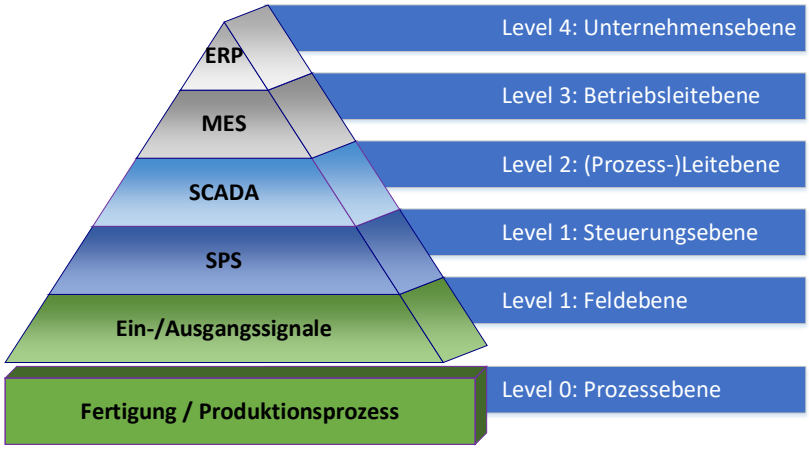


Abbildung 4: Automatisierungspyramide nach [24].

Aufgabe des ERP-Systems ist die Erstellung eines groben Fertigungsplans, den dieses an das MES weiterleitet. Der Fertigungsplan beinhaltet bspw. welche Aufträge wann zu bearbeiten sind. Das MES

übernimmt, basierend auf diesem Fertigungsplan, die detaillierte PPS. Das MES überwacht zudem die Prozessausführung und übermittelt entsprechende Daten an das übergeordnete ERP-System. Neben MES dienen auch *Advanced Planning and Scheduling* (APS) Systeme zur Erweiterung von ERP-Systemen. Die Aufgabe von APS-Systemen besteht in der Simulation, Analyse und Optimierung von unternehmensübergreifenden Geschäftsprozessen über die komplette Wertschöpfungskette. Die Produktionssteuerung gehört nicht zu ihren Aufgaben, was sie von MES unterscheidet [25]. APS-Systeme sind für ihre unterschiedlichen Aufgaben modular gestaltet. Die detaillierte Produktionsplanung findet innerhalb sogenannter *Finite Capacity Scheduling* (FCS) Module statt. FCS-Module führen eine simultane Planung von unterschiedlichen Ressourcen (Material, Maschinen, Werkzeugen oder Personal) durch und bestimmen die Bearbeitungsreihenfolge sowie die Terminierung der Bearbeitungen. Neben diesen rein auf die Produktion bezogenen Systemen, gibt es noch weitere Softwaresysteme, die der Unterstützung und Steuerung unterschiedlicher inner- und gesamtbetrieblicher Prozesse dienen. Beispiele hierfür sind:

- Supply-Chain Management (SCM): *„…bezeichnet den Aufbau und die Verwaltung integrierter Logistikketten (Material- und Informationsflüsse) über den gesamten Wertschöpfungsprozess, ausgehend von der Rohstoffgewinnung über die Veredelungsstufen bis hin zum Endverbraucher.“* [26]
- Total-Quality-Management (TQM): *„Optimierung der Qualität von Produkten und Dienstleistungen eines Unternehmens in allen Funktionsbereichen und auf allen Ebenen durch Mitwirkung aller Mitarbeiter. Total Quality Management strebt die Erhöhung der Kundenzufriedenheit an.“* [27]
- Customer-Relationship-Management (CRM): Strategischer Ansatz zur vollständigen Planung, Steuerung und Durchführung aller interaktiver Prozess mit den Kunden [28].

2.1.2 Dezentrale Produktionssteuerung

Die einzelnen, Maschinen, Produkte sowie andere Fertigungsteilnehmer organisieren sich bei dezentral-heterarchischen, folgend als *dezentral* bezeichneten, Steuerungsansätzen untereinander selbst [10] (siehe Abbildung 3). Eine übergeordnete Steuerungseinheit ist hierzu nicht notwendig. Die Fertigungsteilnehmer müssen miteinander kommunizieren und verhandeln können, um eine funktionierende Selbstorganisation zu erreichen [29]. Diese sind hierzu über Kommunikationsschnittstellen miteinander verbunden. Dezentral Steuerungsarchitekturen bieten eine höhere Flexibilität und eine bessere Reaktionsfähigkeit bei unerwartet auftretenden Ereignissen gegenüber zentralen Ansätzen [11], [30]. Dies ergibt sich dadurch, dass die einzelnen Teilnehmer bei ihrer Entscheidungsfindung wesentlich weniger Elemente mit einbeziehen, als eines zentralen Steuerungssystems, wodurch die Prozesszeit zur Lösungsfindung deutlich geringer ist [10]. Die schnelle Reaktionsfähigkeit ermöglicht eine verbesserte Stabilität des Ablaufplans, da Änderungen zeitnah vorgenommen werden [12].

Der Unterschied zwischen zentralen und dezentralen Steuerungsstrukturen liegt generell darin, dass bei der dezentralen Steuerung lokale Daten zur Entscheidungsfindung herangezogen werden, wohingegen bei der zentralen Steuerung die globale Situation im Rahmen der Entscheidungsfindung Berücksichtigung findet. Dadurch kann bei dezentralen Ansätzen meist keine globale Optimierung erreicht werden [12]. Um dennoch ein gutes Ergebnis zu erreichen, ist es notwendig, die Ziele der einzelnen Teilnehmer gleich auszurichten, damit diese Kollektive das Ziel des Gesamtsystems verfolgen [2]. Ein weiteres Problem bei dezentralen Steuerungsansätzen ist das Fehlen eines Fertigungsplans, der den

Überblick über alle Instanzen beinhaltet [3]. Eine Garantie zur Sicherstellung der spätesten Fertigstellungstermine kann dadurch nicht gegeben werden. Das Einhalten von Lieferzeiten ist bei den oftmals komplexen und globalen Supply-Chains jedoch häufig essenziell für den Unternehmenserfolg.

Eine Möglichkeit zur Realisierung einer dezentralen Steuerungsarchitektur ist die Verwendung autonomer Agenten [31], [32]. Ein Agent kann sowohl ein softwaretechnisches Abbild einer physikalischen Einheit, als auch eine rein virtuelle Einheit sein. Er ist als ein Computersystem, das sich in einer bestimmten Umgebung befindet, in der es autonom handeln kann, um seine Ziele zu erreichen definiert [33]. Agenten beobachten ihre Umwelt durch Sensoren und können den aktuellen Zustand der Umwelt sowie Änderungen innerhalb dieser wahrnehmen. Die dadurch erfassten Informationen werden in der *Informationsverarbeitenden Einheit* (IE) interpretiert und entsprechende Reaktionen und Aktionen geplant. Diese werden anschließend über Aktuatoren ausgeführt, wodurch der Agent seine Umwelt bearbeitet und somit verändert (siehe Abbildung 5).

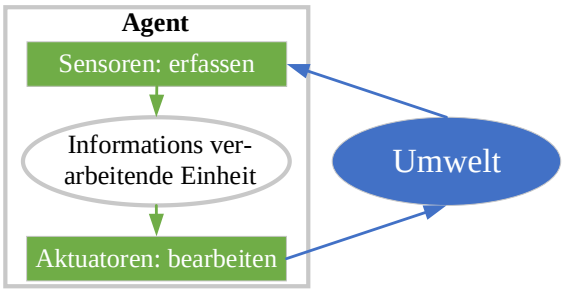


Abbildung 5: Grundstruktur Agent nach [34].

Speziell der Einsatz von Multi-Agenten-Systemen (MAS) wird als vielversprechender Ansatz angesehen, um einem Steuerungssystem die notwendige Flexibilität, Robustheit und Rekonfigurierbarkeit zu ermöglichen [35], [36], [37], [38]. Dieser Ansatz definiert die im Fertigungssystem vorhandenen Fertigungsressourcen als intelligente Agenten, die miteinander kommunizieren und verhandeln, um dynamische Rekonfigurationen durchzuführen und eine hohe Flexibilität zu erreichen [36], [39]. Eine Erläuterung des zur Verhandlung der Agenten untereinander häufig verwenden Kontraktnetzansatzes ist im Anhang M zu finden.

Im Rahmen der Vision *Industrie 4.0* wird erwartet, dass die nächste Generation an Informationssystemen zur Produktionssteuerung agentenbasiert sein wird [40].

2.1.3 Hybride Produktionssteuerung

Neben rein zentralen oder dezentralen Architekturen gibt es auch hybride Ansätze. Diese versuchen die jeweiligen Vorteile miteinander zu kombinieren [41]. Ein Beispiel für eine hybride Architektur ist die in Abbildung 3 gezeigte, dezentrale-hierarchische Struktur. Die Entscheidungsfindung in hybriden Architekturen findet zum einen durch Vorgaben der hierarchisch übergeordneten, zentralen Ebene statt. Die einzelnen Teilnehmer des Fertigungssystems können zum anderen dennoch Entscheidungen zu einem gewissen Grad selbst treffen [2]. In einem solchen hybriden System kann der Ablaufplan zwar von der übergeordneten Einheit kommen, allerdings können die Fertigungsteilnehmer, im Gegensatz zu einem zentralen Ansatz, von diesem Vorschlag abweichen und eine alternative Lösung wählen. Beim Auftreten unerwarteter Ereignisse, ist es durch die Möglichkeit der selbstständigen Entscheidungsfindung der Teilnehmer möglich, schnell eine entsprechende Alternativlösung zu finden.

Anschließend bzw. parallel kann die übergeordnete Instanz einen neuen, global optimierten, Ablaufplan, unter Berücksichtigung der neuen Situation, erstellen [2]. Das System muss durch dieses Vorgehen nicht erst auf eine neue Lösung der übergeordneten Instanz warten, sondern kann in sehr kurzer Zeit, dezentral eine alternative Lösung finden und mit der Fertigung fortfahren.

Meissner et al. [2] haben verschiedene Veröffentlichungen im Bereich der Produktionssteuerung analysiert und haben hierfür zentrale, dezentrale und hybride Ansätze verglichen. Es konnte festgestellt werden, dass hybride Steuerungsarchitekturen oft bessere Ergebnisse bzgl. der Reaktion auf unerwartet auftretende Ereignisse liefern als dezentrale Ansätze. Im folgenden Kapitel werden aktuelle Forschungsarbeiten bzgl. dezentraler Ansätze zur PPS vorgestellt.

2.1.4 Stand der Forschung: Dezentrale Ansätze in der Produktionsplanung und –steuerung

Leusin beschreibt einen MAS-Ansatz [42] für die Echtzeit-PPS in der Werkstattproduktion. Luo et al. stellen einen Ansatz vor, bei dem das komplette MES durch einen MAS modelliert wird, um eine flexible Gestaltung des bestehenden Produktionsprozesses zu realisieren [43]. Merdan et al. haben ein verteiltes, intelligentes Steuerungssystem auf Basis von *Automation Agents* entwickelt, das eine Verbesserung der Reaktion auf Ausfälle oder Überlastungen von Fertigungsressourcen ermöglicht [44]. Vojdani et al. nutzen einen agentenbasierten Ansatz für die detaillierte Produktionsplanung, um die Berechnung von Lieferterminen durch die Verwendung von Echtzeit-Produktionsdaten in der Simulation zu verbessern [45]. Lima et al. stellen ein agentenbasiertes PPS-System vor, das sich dynamisch an Veränderungen im Produktionssystem anpasst [46]. Scholz-Reiter et al. nutzen die Modellierung und Simulation der Produktion auf Basis von Agenten zur Selbstkontrolle von logistischen Prozessen. Das Produkt findet seinen Weg durch das Produktionssystem hierbei selbst [47]. He et al. stellen einen hierarchischen, agentenbasierten Ausschreibungsmechanismus vor, der speziell für die Einzelfertigung konzipiert ist. Dieser arbeitet unter definierten Randbedingungen und stellt sicher, dass die Ressourcen zur Erfüllung von Kundenaufträgen selbstorganisiert und kostengünstig eingesetzt werden [48]. Cupek et al. stellen ein agentenbasiertes MES vor, das speziell auf die Bedürfnisse der Produktion von kleinen Batchgrößen spezialisiert ist [49]. Das vorgeschlagene System ist komplett heterarchisch organisiert und hat somit keine zentrale Organisationseinheit. Die Umsetzung des Ansatzes erfolgt unter der Verwendung von Agenten, welche Produktionsdaten sammeln und einen Teil des cyberphysischen Systems darstellen. Klein et al. [50] stellen einen agentenbasierten Ansatz zur Produktionsablaufplanung vor, welcher auf dem Verhandeln zwischen den einzelnen Agenten basiert. Die Entscheidungsfindung zur Erstellung des Ablaufplans findet komplett dezentral statt und das Produkt sucht sich seinen Weg durch die Fertigung selbst. Das Produkt selbst stößt bei den verfügbaren Ressourcen die Bearbeitung einer Operation an, worauf diese dem Produkt Angebote unterbreiten. Anschließend wählt das Produkt die Ressource mit dem besten Angebot zur Durchführung der Operation aus. Das MAS iAgent [51] wurde zur Steuerung einer Abfüllanlage entwickelt und unter Verwendung des Java Frameworks JADE realisiert. Bei dem vorgestellten System handelt es sich um ein komplett dezentrales Steuerungssystem. Große Erdmann [52] stellt ein autonomes Kontrollsystem für die Optimierung der Intralogistik vor, welches als MAS realisiert ist. Für den Transport werden neben FTS auch andere Transportmittel berücksichtigt. Ausgelegt ist das System für die Flow-Shop-Produktion und nutzt eine Kombination aus GA und Nachbarschaftssuche zur Optimierung der Zuordnung der notwendigen Transportaufträge zu den vorhandenen Transportmitteln. Das System ist zudem in der Lage auf neueintreffende Aufträge autonom zu reagieren. Die Maschinenbelegung ist durch die Flow-

Shop-Umgebung fest definiert und wird somit in der Ablaufplanung nicht berücksichtigt. Das Kontrollsystem steuert die Steuerung der Transportmittel, nicht aber der Fertigungsressourcen. Cardin et al. [53] haben in ihrer Analyse bzgl. aktueller Agenten-Architekturen zur Steuerung von Produktionssystemen festgestellt, dass es meist keine Möglichkeit der Integration des Menschen in die Lösungsfindung gibt. Hierdurch könnten allerdings die Vorteile des Menschen und die von Agenten kombiniert werden, weshalb Cardin et al. dieses Feld als notwendige Herausforderung zur einfachen und sicheren industriellen Integration solcher Architekturen ansehen.

Keiner der aufgeführten Ansätze berücksichtigt sowohl die Steuerung der Fertigungsressourcen als auch die Integration und Steuerung eines flexiblen Materialtransports mittels FTS. Des Weiteren werden die speziellen Herausforderungen der Refabrikation nicht berücksichtigt oder abgebildet. So gibt es keine Möglichkeit, auf Inspektionsergebnissen basierende Änderungen des Prozessplans eines Produktes während der Prozessausführung in die Produktionssteuerung einfließen zu lassen. Eine hierfür oftmals notwendige Integration des Menschen, welcher basierend auf seinem Expertenwissen Entscheidungen über den weiteren Prozessplan eines Produktes trifft und diese Entscheidung an das System weitergeben muss, ist in den vorgestellten Ansätzen ebenfalls nicht vorgesehen.

Nachdem in den vorangegangenen Kapiteln die Aufgaben der PPS sowie die unterschiedlichen Steuerungsarchitekturen erläutert und der aktuellen Stands der Forschung zu dezentralen Ansätze in der PPS gegeben wurden, erfolgt im folgenden Kapitel die Vorstellung besonderen Ansprüche der Refabrikation an die PPS.

2.1.5 Anforderungen an die Produktionsplanung und -steuerung in der Refabrikation

Die Refabrikation eines Produktes besteht aus den sechs, in Abbildung 6 dargestellten Prozessschritten [54]. Die zerstörungsfreie Demontage eines Produktes ist hierbei die notwendige Ausgangsbasis. Eine Demontage ist das Zerlegen eines technischen Komplexes, welches aus mehreren miteinander verbundenen Einzelteilen besteht. Am Ende der Demontage sollen alle ausgewählten, sich innerhalb des zu demontierenden technischen Komplexes befindlichen, Bauteile einzeln vorliegen. Die demontierten Bauteile werden anschließend gereinigt und später einer Inspektion unterzogen. Basierend auf den Ergebnissen dieser Inspektion erfolgt die Sortierung der Bauteile. Diese werden, je nachdem ob und wie sie überholt werden müssen, sortiert. Bauteile, bei denen eine Überholung nicht mehr möglich ist, werden ersetzt. Die erneute Montage des Produktes erfolgt, sobald die entsprechenden Teile ersetzt oder überholt sind. Als abschließender Prozessschritt wird das Produkt einem Funktionstest unterzogen. Verläuft dieser erfolgreich, wird das Produkt mit einer erneuten Garantie versehen. Dieses kann anschließend einen weiteren Lebenszyklus durchlaufen. Die Refabrikationsprozessschritte sind je nach Produkt unterschiedlich und können von dem aufgezeigten Prozessablauf abweichen.

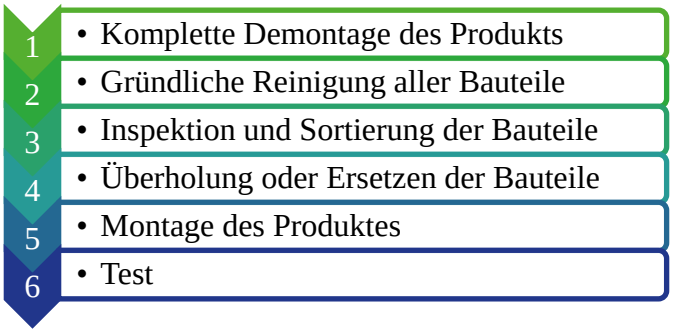


Abbildung 6: Prozessablauf einer Refabrikation [54].

Die Refabrikation stellt besondere, aus dem Bereich der Fertigung nicht bekannte, Herausforderungen an die PPS [55]. Beispiele hierfür sind [56]–[61]:

- Unbekannter Produktzustand der zu refabrizierenden Produkte durch unterschiedliche Beanspruchung während des vorherigen Einsatzes;
- Variierende Bearbeitungszeiten;
- Balance zwischen Kundennachfrage und Rückgabe von Gebrauchtprodukten;
- Nicht deterministisches Routing für Materialien und Produkte durch das RS;
- Unbekannte Ankunftszeit und Menge der gebrauchten Produkte.

Dadurch unterscheidet sich auch das Layout eines RS von dem einer klassischen Fertigungslinie. Dieser Layoutunterschied ist in Abbildung 7 dargestellt. Die mit *Sx* beschriebenen Kreise stellen die vorhandenen Bearbeitungsstationen dar. Gleichfarbige Pfeile zeigen den Weg eines bestimmten Produktes durch das System. Alle Produkte eines Produkttyps nehmen in der traditionellen Fertigungslinie dieselbe Route durch das Fertigungssystem. Deshalb kann der Materialtransport hier durch starre Transportsysteme, wie Förderbänder, realisiert werden. In der Refabrikation können Produkte eines Produkttyps unterschiedliche Routen durch das RS nehmen. Produkt A1 und Produkt A2 sind beide Produkte vom Typ A, weisen allerdings unterschiedliche Beschädigungen auf. Dadurch sind für die jeweilige Refabrikation individuelle Prozessschritte notwendig und beide Produkte müssen verschiedene Routen durch das RS nehmen. Infolgedessen wird im RS ein flexibler Materialtransport benötigt. Der Begriff Flexibilität kann als die Fähigkeit definiert werden, auf Umweltveränderungen zu reagieren und sich an diese anzupassen zu können. Des Weiteren können die unbekannten Zustände der demontierten Teile dazu führen, dass es bei den geplanten Refabrikationsprozessschritten zu Komplikationen kommt und eine andere Kombination von Refabrikationsprozessschritten gewählt werden muss. Dies hat eine erneute Ausführung der Ablaufplanung (Rescheduling) zur Folge. Ein einfaches Beispiel hierfür ist eine defekte Schraube, welche sich nicht wie geplant automatisiert lösen lässt, sondern einer manuellen Nacharbeit unterzogen werden muss. Die Neuplanung muss in möglichst kurzer Zeit erfolgen, um Stillstände des RS zu verhindern, oder zumindest zu reduzieren. Deshalb ist die Verwendung eines schnellen Optimierungsalgorithmus für die Ablaufplanung von RS notwendig.

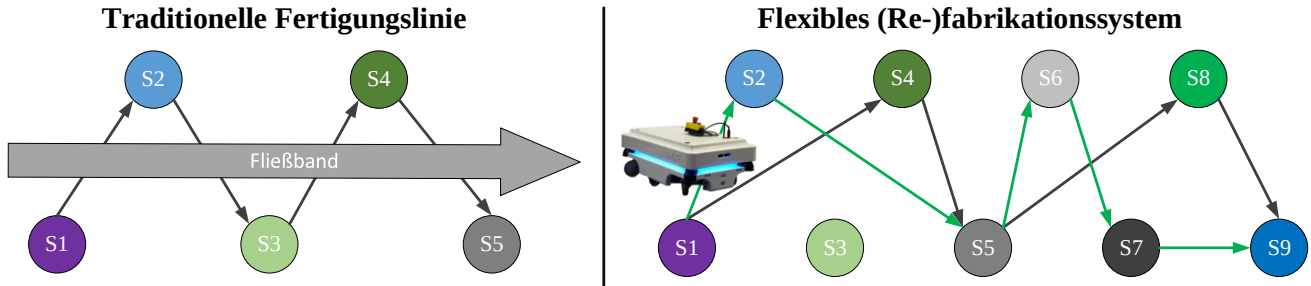


Abbildung 7: Unterschied zwischen traditioneller Fertigungslinie und flexiblem RS.

Im Anhang F ist eine detaillierte Beschreibung der ökologischen und ökonomischen Motivationspunkte zur Durchführung einer Refabrikation zu finden.

2.1.6 Spezielle Ansätze zur Produktionsplanung und –steuerung in der Refabrikation

Die Forschungsaktivitäten zur PPS im Bereich der Refabrikation konzentrieren sich oft auf die Planung und Steuerung der Demontage. Junior und Filho [62] stellen bspw. ein dynamisches Programmiermodell zur Planung der Demontage vor. Das vorgeschlagene Modell wird anhand der Refabrikation einer Automobil-Kupplung getestet und validiert. Ehm [63] stellt einen auf Mixed Integer Programmierung (MIP) basierenden Ansatz zur Demontageablaufplanung vor. Das MIP basiert auf einem Demontageprozessdiagramm, welches sowohl parallele als auch alternative Operationen darstellt. Eine industrielle Fallstudie wird zur Demonstration der Anwendbarkeit des Ansatzes vorgestellt. Problem bei der Verwendung von MIP ist allerdings der sehr hohe Rechenaufwand, was diese Methodik zur Steuerung der Demontage ungeeignet macht. Jungbluth, Gerke und Plapper [64] stellen einen intelligenten Roboterassistenten für die zerstörungsfreie Demontage vor. Der vorgeschlagene Ansatz führt die Demontageplanung auf Basis eines erweiterten CAD-Produktmodells autonom durch. Das System allokiert zudem die einzelnen Aufgaben zwischen Mensch und Roboter und passt den Demontageplan beim Auftreten unerwarteter Ereignisse automatisch an.

Forschungsaktivitäten, die sich mit der PPS des gesamten Refabrikationsprozesses beschäftigen werden in diesem Abschnitt vorgestellt. He [65] präsentiert einen hybriden Algorithmus, der ein neuronales Backpropagationsnetzwerk mit einem genetischen Algorithmus (GA) kombiniert. Ziel ist Optimierung der Produktionsplanung eines RS unter Berücksichtigung von Unsicherheiten. Die Validierung des Ansatzes erfolgt durch Simulationsstudien. Kim et al. [66] nutzen ein gemischt ganzzahliges Programmiermodell sowie einen auf Prioritätsregeln basierenden Planungsansatz für die PPS einer Refabrikation. Die Validierung des Ansatzes erfolgt sowohl anhand von Simulationsstudien als auch durch die prototypische Implementierung innerhalb einer Refabrikations-Modellfabrik. Wen et al. [67] stellen einen hybriden Algorithmus mit bi-randomer Simulationstechnik, neuronalem Netzwerk und GA vor, um die PPS in einem RS mit Unsicherheiten durchzuführen und zu optimieren. Der Ansatz wird anhand einer Fallstudie getestet und validiert. Cui et al. [55] untersuchen Ablaufplanungsprobleme in der Domäne Refabrikation. Mehrere Merkmale, welche die Ablaufplanung in der Refabrikation gegenüber der Fertigung komplexer gestalten, wurden identifiziert und untersucht. Ein Lösungsansatz wird allerdings nicht präsentiert. Qui et al. [68] stellen einen MILP-basierten Optimierungsansatz für die Supply-Chain-Logistik in der Refabrikation vor. Liu et al. [69] präsentieren ein Optimierungsverfahren für die Steuerung von Wiedermontageprozessen innerhalb einer Refabrikation. Das Optimierungsverfahren kombiniert ein Zustandsmodell mit dynamischer Programmierung. Verifiziert wird das Verfahren in der Wiedermontagelinie einer Kurbelwellen-Refabrikation. Die Qualität der refabrizierten Produkte kann hierbei um 5,27 % verbessert und die Kosten des Montagesystems um 2,76 % senkt werden. Brusaferri et al. [70] nutzen eine Unified Particle Swarm Optimization als On-Line Ablaufplanungsansatz in der Refabrikation von gedruckten Leiterplatten. Die Zielfunktion kombiniert die gewichtete Gesamtverzögerung sowie die Leerlaufzeit der Maschinen, um den Energieverbrauch aufgrund von Dissipationen zu minimieren. Validiert wird der Ansatz durch dessen Implementierung in einer Modellfabrik, wo eine Verbesserung gegenüber dem zuvor eingesetzten Ablaufplanungsverfahren nachgewiesen werden kann. Gao et al. [71] nutzten einen Discrete Harmony Search (DHS) Algorithmus für Ablaufplanungs- sowie Neuplanungsaufgaben in der Refabrikation. Die vorgestellte Neuplanungsstrategie wird zur Berücksichtigung unerwartet langer Prozesszeiten sowie zusätzlich hinzugekommener Montageprozessschritte angewandt. Simulationsstudien und Vergleiche zeigen, dass die vorgeschlagene DHS in der Lage ist, die Ablaufplanungs- und Neuplanungsprobleme effektiv zu lö-

sen. Gong et al. [72] schlagen in Ihrer Arbeit ein mathematisches Modell zur integrierten und simultanen Prozess- und Ablaufplanung für die Refabrikation vor. Ein Hybrid Multi-Objective Evolutionary Algorithm (HMEA) wird zur Lösung des Modells genutzt. Die Verifikation erfolgt anhand von Simulationsstudien, in denen der HMEA mit anderen Algorithmen vergleichen wird. Der vorgeschlagene HMEA ist in der Lage, mehr und bessere Pareto-Lösungen für das vorliegende Problem zu finden. Zhang [73] nutzt einen hybriden Optimierungsalgorithmus bestehend aus Fuzzy Algorithm, Backpropagation (BP) Neural Network und GA zur Ablaufplanung in RS unter der Berücksichtigung unbekannter Faktoren. Als unbekannte Faktoren werden die Zustände verschiedener Bauteile eines zu refabrizierenden PKW-Verbrennungsmotors herangezogen. Die erzielten Simulationsergebnisse zeigen, dass der hybride Algorithmus eine gute Konvergenz erreicht und die erhaltene Lösung die Gesamtkosten für die Ablaufplanung und die Bearbeitungszeit minimieren kann. Ein Vergleich zu anderen Algorithmen und Verfahren erfolgt allerdings nicht. Zur Optimierung der Ablaufplanung einer Batch-Refabrikation definieren Li et al. [74] diese als ein Flexible Job-Shop Scheduling Problem (FJSSP) und modellieren das Problem als MILP-Modell. Als Zielfunktion dient die Minimierung der Stromkosten sowie der DLZ. Gao et al. [75] präsentieren einen bi-objektiven Water Cycle Algorithm (WCA) zur Einplanung neuer Aufträge innerhalb der Refabrikationsneuplanung. Sechs reale Refabrikationsfälle mit unterschiedlichen Maßstäben werden hierzu untersucht und für die Lösung dieser Fälle der vorgeschlagene WCA dem ebenfalls bi-objektiven Algorithmus, NSGAII gegenübergestellt. Der WCA liefert bzgl. der verwendeten Zielfunktion, bessere Ergebnisse als der NSGAII. Yu und Le [76] definieren in Ihrer Arbeit ein neues Ablaufplanungsproblem für RS mit parallelen Demontagearbeitsplätzen, einer Job-Shop Wiederaufarbeitungswerkstatt sowie parallelen Wiedermontagearbeitsplätzen. Diese Problemstellung wird als ganzzahliges Programmiermodell beschrieben. Die Optimierung der Ablaufplanung erfolgt anhand zweier Lösungsansätze, wobei ein Ansatz die drei Stationen als sequenzielle Probleme, und der andere alle drei Stationen als ein simultanes Problem betrachtet. Die vorgestellten Simulationsstudien zeigen, dass die Ergebnisse des simultanen Ansatzes die des sequenziellen Ansatzes deutlich übertreffen. Li et al. [77] abstrahieren ein RS als Job-Shop Scheduling Problem (JSSP) und betrachten hier insbesondere variierende Prozesszeiten, Ressourcenkonflikte und alternativ Prozessrouten zur Refabrikation eines Produktes. Ein Colored Timed Petri Net (CTPN) wird zur Modellierung des Problems genutzt. Auf das CTPN wird eine modifizierte Variante der Kombination aus Simulated Annealing (SA) und Minimum Slack Time (MST) zur Optimierung des Problems angewendet. Als Zielfunktion wird die Minimierung der Produktionskosten verwendet. Der Ansatz wird zwei weiteren Optimierungsmethoden gegenübergestellt, wobei eine fixe Prozessrouten verwendet und die zweite Methode die Standardvariante von SA / MST, anstatt der vorgestellten, modifizierten Variante, nutzt. Die Überlegenheit des vorgestellten Ansatzes gegenüber beiden Alternativen kann aufgezeigt werden. Über die Möglichkeit zur Reaktion auf unerwartet eintretende Ereignisse verfügt der Ansatz hingegen nicht. Dies wird aber als ein in Zukunft wichtiger, zu betrachtender Aspekt aufgeführt.

Keiner der oben aufgeführten Ansätze berücksichtigt die Integration eines flexiblen Materialtransports in der Ablaufplanung des RS. Die Reaktion auf unerwartet eintretende Ereignisse wird zudem in wenigen Studien berücksichtigt. Maschinenausfälle oder die Integration neuer notwendiger Operationen, welche aus Inspektionsprozessen während der Prozessausführung resultieren, und im Rahmen einer Neuplanung mit in die Lösungsfindung einzubinden sind, werden ebenfalls in keiner der Arbeiten berücksichtigt.

Nach der Vorstellung der Besonderheiten der Refabrikation im Rahmen der PPS sowie der Präsentation des aktuellen Stands der Forschung in diesem Bereich wird im folgenden Kapitel das Thema Feinplanung erläutert und verschiedene Optimierungsalgorithmen hierzu vorgestellt.

2.2 Feinplanung

Die Feinplanung ist ein Teilbereich der Produktionssteuerung und besitzt zwei hauptsächliche Aufgaben. Sie legt zum einen fest, welche der Ressourcen die einzelnen Produktionsaufträge bearbeiten und zum anderen wird die zugehörige Reihenfolgeplanung durchgeführt. Dieser Schritt wird häufig auch als Feinterminierung oder Ablaufplanung (engl. Scheduling) bezeichnet [78]. Der Planungshorizont ist kurzfristig ausgelegt und beträgt in der Regel weniger als drei Tage. Im Rahmen der Reihenfolgeplanung werden in kommerziell erhältlichen Softwaresystemen häufig einfach heuristische Verfahren unter der Nutzung von Prioritätsregeln verwendet [3], [78]. Eine genaue Beschreibung von Prioritätsregeln und anderen Optimierungsverfahren zur Ablaufplanung erfolgt an späterer Stelle. In industriellen Betrieben sind kommerzielle Softwaretools zur Ablaufplanung noch nicht flächendeckend verbreitet. Selbst entwickelte Software, papierbasierte Lösungen oder Microsoft Office-Tools sind hier oftmals vorzufinden [79].

Neben der Optimierungsqualität ist auch die Reaktionsfähigkeit des verwendeten Optimierungsalgorithmus bei der Ablaufplanung von Bedeutung. Ein Optimierungsalgorithmus ist reaktionsfähig, wenn er bei Änderungen an der vorliegenden Problemstellung in der Lage ist, innerhalb kurzer Zeit einen neuen, angepassten sowie gültigen Ablaufplan zu erstellten. Ein Fertigungsplan ist gültig, wenn sowohl die Reihenfolgeabhängigkeiten der einzelnen Operationen innerhalb eines Auftrags nicht verletzt sind, als auch die Kapazitäten der Ressourcen nicht überschritten werden. Änderung an der Problemstellung können sich durch unerwartet auftretenden Ergebnisse während der Ausführung des Ablaufplans ergeben. Benötigt der Optimierungsalgorithmus für die Erstellung einer neuen Lösung zu lang, kann dies zu Fertigungsstillständen führen. Die Erfassung und Auswertung von Maschinen- und Sensordaten ist zur Anpassung der Problemstellung an die neuen Randbedingungen notwendig [30].

2.2.1 Ablaufplanungsprobleme

Ablaufplanungsprobleme treten in allen Wirtschaftsbereichen auf, von der Netzwerktechnik, über den Bahnverkehr bis hin zur Fertigung. Sie gehören der Klasse der kombinatorischen Optimierungsprobleme an. Zur allgemeinen Abbildung verschiedener Arten von Fertigungssystemen als Ablaufplanungsprobleme, gibt es allgemeine Definitionen dieser, wie bspw. Flow-Shop, Job-Shop, Flexible Job-Shop [80]–[88] oder die Re-entrant Fertigungssysteme [34], [37], [38], [42],[34], [89]. Diese unterschiedlichen Definitionen versuchen, die jeweiligen Charaktere und Besonderheiten abzubilden. Ein RS kann, mit einigen Anpassungen, durch ein JSSP bzw. FJSSP abgebildet werden (Kapitel 2.2.3), weshalb diese beiden Arten von Fertigungssystemen folgend genauer erläutert werden.

Das Ziel des JSSP besteht darin, einen zeitlichen Ablauf für die einzelnen Operationen der vorliegenden n Aufträge zu erstellen. Ein Auftrag besteht aus mehreren Operationen, welche zur Fertigung eines Produktes durchzuführen sind. Die einzelnen Operationen werden auf vorher zugeordneten Maschinen durchgeführt. Die technologischen Einschränkungen spezifizieren für jeden Auftrag eine feste, einzuhaltende Operationsreihenfolge. Diese sind im Voraus bekannt. Jede Maschine ist von Zeitpunkt null verfügbar und die Operationen werden ohne Unterbrechung ausgeführt. Die Minimierung der DLZ wird als Zielfunktion verwendet. Die DLZ (engl.: *makespan / lead time*) ist die Zeit, die vom Beginn

der Bearbeitung des ersten Auftrags bis zur Fertigstellung des letzten Auftrags vergeht. In [90] wird die DLZ wie folgt definiert: *„Die Durchlaufzeit eines Produktionsauftrages bezeichnet die Zeitspanne vom Beginn der Bearbeitung des Arbeitsvorgangs bis zur Fertigstellung der letzten Arbeitsfolgen. Sie kann aber auch für einzelne Arbeitsvorgänge definiert und ermittelt werden“*. Die DLZ besteht somit aus der Summe von Bearbeitungs-, Warte-, Transport- und Liegezeiten. Eine Reduzierung der DLZ dient der Senkung der durchschnittlichen Kapitalbindung sowie der gleichzeitigen Erhöhung der Liefertreue und –flexibilität [90]. Die DLZ kann auch für die Fertigung eines einzigen Produktes betrachtet werden. Neben der DLZ kann das JSSP auch hinsichtlich anderer Optimierungsfunktionen wie bspw. der Maximierung der Auslastung der Maschinen (engl. OEE – Overall Equipment Effectiveness), oder auch multikriteriell optimiert werden. JSSP in ihren statischen und deterministischen Formen sind einfach zu beschreiben, aber schwer zu lösen, da es sich um NP-schwere Problem handelt [91]. NP steht für *nicht deterministische Polynomialzeit* und ist eine Komplexitätsklasse, welche eine Eigenschaft eines algorithmischen Problems bezeichnet. Die Rechenzeit (RZ) steigt bei Problemen dieser Komplexitätsklasse mit steigender Problemgröße exponentiell. Eine genaue Beschreibung von NP-schwer und der Komplexität kombinatorischer Optimierungsprobleme ist im Anhang G zu finden.

Ein JSSP-Beispiel ist in Abbildung 8 aufgeführt. Dieses umfasst drei Aufträge (Zeilen innerhalb der Tabelle), bestehend aus jeweils zwei bis drei Operationen (Spalten), die auf drei verfügbaren Maschinen (*M1*, *M2*, *M3*) bearbeitet werden sollen. Die Werte innerhalb der Klammern hinter den jeweiligen Maschinen entsprechen der Prozesszeit der zugehörigen Operation auf dieser Maschine. Als Lösung des JSSP liegt am Ende ein gültiger Ablaufplan vor. Dieser kann als Gantt-Diagramm grafisch dargestellt werden. Auf der Ordinatenachse werden die verfügbaren Maschinen und auf der Abszissenachse die Zeit dargestellt. Die einzelnen Operationen der Aufträge werden im Gantt-Diagramm in der Zeile der zugeordneten Maschine mit den jeweiligen Start- und Endzeiten als Blöcke eingetragen. Das Gantt-Diagramm in Abbildung 8 stellt einen gültigen Fertigungsplan des JSSP-Beispiels dar. Folgend wird die Interpretation des Gantt-Diagramms erläutert. Die zweite Operation des dritten Auftrags O_{32} ist in Abbildung 8 Maschine M_3 zugeordnet. Der Startzeitpunkt ist S_{32} und entsprechend der Prozesszeit von P_{32} auf dieser Maschine, ergibt sich eine Endzeit von E_{32}. t_0 entspricht entweder Zeitpunkt $t_0 = 0$ bei der initialen Planung, oder, wenn eine Neuplanung durchgeführt wird, dem aktuellen Zeitpunkt.

	Operationen		
Aufträge	1	2	3
Auftrag 1	M1 (16)	M2 (8)	M3 (4)
Auftrag 2	M3 (9)	M2 (9)	
Auftrag 3	M2 (7)	M3 (10)	M1 (12)

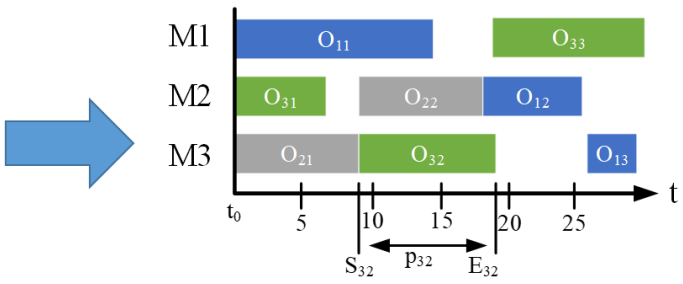


Abbildung 8: JSSP mit drei Auftragen und drei Maschinen sowie zugehöriger, gültiger Ablaufplan als Gantt-Diagramm.

Neben dem allgemeinem JSSP gibt es auch noch einige Erweiterung dieses Problems, wie bspw. das FJSSP, welches im Folgenden genauer erläutert wird.

2.2.2 Das flexible Job-Shop Scheduling Problem

In realen Fertigungssystemen sind häufig parallele Maschinen vorzufinden [92], [93]. Als parallele Maschinen gelten mehrere Maschinen, die über gleiche eine Funktionalität verfügen. Dadurch dass es sich nicht um exakt gleiche Maschinen handeln muss, kann es dazu kommen, dass parallele Maschinen

zwar die gleiche Funktionalität bieten, für die Durchführung einer Operation allerdings unterschiedliche Prozesszeiten benötigen. Beispielhaft hierfür sind zwei CNC-Fräsen von unterschiedlichen Herstellern. Diese können die gleichen Operationen durchführen, brauchen hierfür aber evtl. unterschiedliche lange Prozesszeiten. Auch verschiedene Verfahren bspw. das Fräsen oder alternativ der 3D-Druck eines Bauteils können durch parallele Maschine dargestellt werden. Zur Abbildung von parallelen (alternativen) Maschinen wurde in den 1970er Jahren das generalisierte FJSSP definiert [94]. Das FJSSP ist eine dahingehende Erweiterung des JSSP, dass eine Operation auf verschiedenen, alternativen Maschinen bearbeitet werden kann. Die Flexibilität innerhalb dieser Problembeschreibung wird von Hutchinson und Pflughoeft [95] als sogenannte *Maschinenflexibilität* bezeichnet. Diese wird so definiert, dass jede Maschine eine Menge an Operationen spezifiziert, welche auf ihr durchgeführt werden können. Das Ziel des FJSSP ist es, einen gültigen Ablaufplan zu finden, der die DLZ für eine Reihe von Aufträgen und Operationen auf Mehrzweckmaschinen minimiert.

Das FJSSP besteht aus zwei Teilproblemen. Ein Teilproblem besteht in der Zuordnung jeder Operation zu einer Maschine aus einem Satz von Maschinen, welche die entsprechende Operation durchführen können. Das zweite Teilproblem besteht darin, die zugeordneten Operationen auf allen Maschinen zu sequenzieren, um einen gültigen Ablaufplan zu erhalten, der eine vordefinierte Zielfunktion minimiert. Das FJSSP ist durch das zusätzliche Teilproblem der Maschinenzuordnung eine komplexere Version des JSSP. Das FJSSP ist stark NP-schwer [94].

In Abbildung 9 ist ein FJSSP-Beispiel in Tabellenform dargestellt. Dieses besteht aus zwei Aufträgen und drei Maschinen. Der erste Auftrag beinhaltet drei und der zweite Auftrag zwei auszuführende Operationen. Die Werte innerhalb der Klammern hinter den jeweiligen Maschinen *M1,…,M3* geben die Prozesszeiten an. Jede Zeile beinhaltet, wenn vorhanden, eine mögliche Maschine zur Bearbeitung der jeweiligen Operation. Die zweite Operation von Auftrag 2, $O_{2,2}$, kann bspw. auf Maschine *M2*, oder *M3* ausgeführt werden.

	Operationen		
Aufträge	1	2	3
Auftrag 1	M1 (10)	M1 (25)	M1 (30)
	M3 (25)		M2 (15)
			M3 (25)
Auftrag 2	M1 (10)	M2 (12)	
	M2 (15)	M3 (18)	

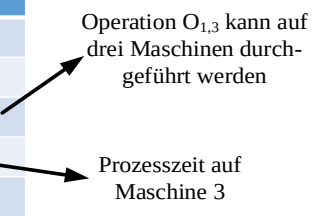


Abbildung 9: FJSSP mit zwei Aufträgen und drei Maschinen (M1-M3).

2.2.3 Definition eines (Re-)fabrikationssystems als Flexible Job-Shop Scheduling Problem

Der generelle Aufbau eines RS wurde bereits in Abbildung 7 dargestellt und ist in Abbildung 10 nochmals mit der Untergliederung, Demontage, Refabrikation und Remontage, nach [60] aufgeführt. Der Bereich Refabrikation besteht aus mehreren Arbeitsstationen (in Abbildung 10 mit *M* für Maschinen bezeichnet), welche von den zu refabrizierenden Produkten unterschiedlich durchlaufen werden. Es handelt sich also um ein flexibles System, das einen ebenfalls flexiblen Materialtransport benötigt. Dieses kann teilweise durch ein JSSP, bzw. um auch alternative Maschinen darzustellen, durch ein FJSSP beschrieben werden.

Neben automatisierten und spezialisierten Arbeitsplätzen, an denen ein bestimmter Arbeitsgang ausgeführt werden kann, gibt es in RS oft auch manuelle Arbeitsplätze. Der Werker kann an diesen verschiedene Operationen durchführen. Diese Arbeitsplätze werden häufig für die Behebung von Defekten an dem gebrauchten Produkt genutzt, welche durch die vorhandenen automatisierten oder spezialisierten Prozesse nicht behoben werden können. Auch evtl. notwendige Nacharbeiten finden auf diesen Arbeitsplätzen statt. Die Problembeschreibung des RS als FJSSP erlaubt es in diesem Zusammenhang zudem, die folgenden Maschinentypen darzustellen:

- Spezifische Maschinen für nur eine Operation
- Parallele spezifische Maschinen für nur eine Operation
- Maschinen, die für mehrere Operationen geeignet sind
- Parallele Maschinen, die für mehrere Operationen geeignet sind

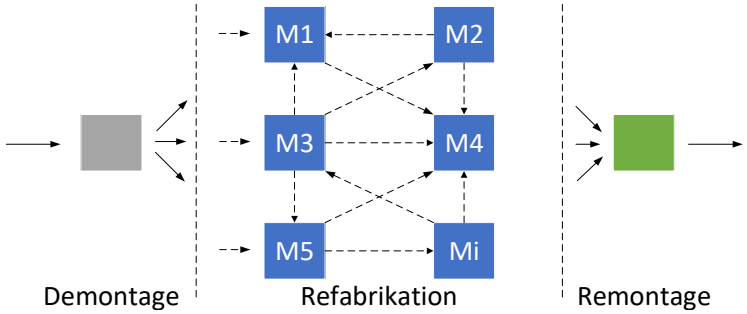


Abbildung 10: Darstellung eines RS nach [60].

Allerdings werden nicht alle Randbedingungen eines RS in einem FJSSP berücksichtigt. Die zu refabrizierenden Produkte nehmen, je nach Defekten unterschiedliche Routen durch das RS, weshalb ein flexibler Materialtransport notwendig ist. Das klassische FJSSP beinhaltet allerdings keine Berücksichtigung des Transports der Produkte bzw. Aufträge durch das System. Deshalb wird in Kapitel 2.4.2 eine Erweiterung des FJSSP, die SAMF, vorgestellt.

2.2.4 Dynamische Ablaufplanungsprobleme

In Fertigungssystemen treten häufig unerwartete Ereignisse, wie neue Aufträge oder Maschinen- bzw. Transportmittelausfälle, auf. In der Refabrikation kommen zudem, sich während der Prozessausführung ändernde Prozesspläne, hinzu. Diese resultieren aus Inspektionsprozessen der Produkte bei denen zusätzlich notwendige Prozessschritte festgestellt werden. Das JSSP und FJSSP stellen jedoch deterministische Aufgaben dar. Unerwartet auftretende Ereignissen werden dabei nicht berücksichtigt, was für die Steuerung eines RS unzureichend ist [96]. Nur wenige Fallstudien oder praktische Anwendungen betrachten Ablaufplanungsprobleme innerhalb nicht deterministischer Umgebungen, wodurch es in diesem Bereich Forschungsbedarf gibt [97].

Tritt eine unerwartete Verzögerung bei der Bearbeitung einer Operation auf, können die folgenden Operationen auf dieser Maschine ihre geplanten Start- und Endzeitpunkte nicht mehr einhalten. Dies betrifft auch alle Nachfolgeoperationen der betroffenen Aufträge. Ein Maschinenausfall führt dazu, dass die darauf geplanten Operationen nicht mehr auf dieser bearbeitet werden können und auf eine andere Maschine umgeplant werden müssen. Eine Kettenreaktion ist die Folge, da von der Umplanung dieser Operationen auch alle nachfolgenden Operationen der zugehörigen Aufträge betroffen sind. Die Umplanung der Operationen auf alternative Maschinen bewirkt zudem, dass auch die auf diesen Ma-

schinen geplanten Operationen und deren Nachfolgeoperationen betroffen sind. Durch diese Kettenreaktion kann bereits eine leichte Änderung dazu führen, dass der Ablaufplan undurchführbar wird [98]. Um die Fertigung dennoch aufrecht zu erhalten, werden oftmals direkt auf der Fertigungsebene nicht optimale Entscheidungen bzgl. der Ablaufplanung getroffen und angestoßen. „Online“ Ablaufplanungsverfahren hingegen können beim Auftreten unerwarteter Ereignisse gültige und dennoch optimierte Ablaufpläne liefern. Diese Verfahren können in Neuplanungs- und kontinuierliche Anpassungsstrategien untergliedert werden.

Neuplanungsstrategien erstellen nach dem Eintreten eines unerwarteten Ergebnisses, unter Berücksichtigung der aktuellen Situation auf der Fertigungsebene, einen neuen Ablaufplanplan. Hierfür werden oft einfache Konstruktionsverfahren verwendet, da diese aufgrund ihrer kurzen RZ schnell einen angepassten, gültigen Ablaufplan generieren können [99]. Nachteil dieser Verfahren ist, dass bei häufig unerwartet eintretenden Ereignissen und der damit einhergehenden Erstellung von ständig neuen Ablaufplänen, Unruhe in der Fertigung entsteht. Dies zieht verschiedene negative Auswirkungen mit sich [100]. Eine einfache Möglichkeit dieser Problematik entgegenzuwirken, ist es, bei der Erstellung des Ablaufplans Pufferzeiten zwischen den einzelnen aufeinanderfolgenden Operationen einzuplanen. Diese Pufferzeiten verringern die Wahrscheinlichkeit, dass die Verspätung einer Operation zu einer Veränderung der Startzeit der nachfolgenden Operation führt. Eine weitere ähnliche Möglichkeit, der Häufigkeit der Erstellung neuer Ablaufpläne entgegenzuwirken, ist es, die Bearbeitungsdauer der einzelnen Operationen um die Ausfallrate der Maschine, auf welcher sie bearbeitet werden, künstlich zu verlängern. Durch dieses Vorgehen wird nicht die realitätsferne Annahme einer immer gegebenen Verfügbarkeit der Ressourcen getroffen, sondern eine realitätsnähere Annahme getroffen. Die Ausführbarkeit des Ablaufplänen wird dadurch wahrscheinlicher [101]. Konträr dazu steht die einhergehende, längere und suboptimale DLZ. Auf diese Weise erstellte Ablaufpläne werden als *robuste Fertigungspläne* bezeichnet [102]. Ein weiteres Vorgehen, die Anzahl der Neuplanungsvorgänge zu reduzieren, ist es, die Neuplanung in definierten Zeittakten (bspw. jede halbe Stunde) periodisch durchzuführen. Die Neuplanung kann auch bspw. dann ausgeführt werden, wenn eine vorher definierte Abweichung bzgl. des eigentlichen Ablaufplans überschritten wurde [103]. Im Gegensatz zu den robusten Fertigungsplänen besitzen diese Verfahren eine höhere Planungsgüte. Dies geht allerdings damit einher, dass die Fertigungspläne mit höherer Wahrscheinlichkeit nicht wie geplant umgesetzt werden können. Auch die Kombination aus robuster und periodischer bzw. abweichungsabhängiger Ablaufplanung ist möglich [104].

Neben der Neuplanung ist auch die Anpassung des aktuellen Ablaufplans an die neue Situation eine weitere Möglichkeit auf unerwartet eintretende Ereignisse zu reagieren. Heuristische Verbesserungsverfahren können in diesem Fall den Ablaufplan soweit anpassen, bis dieser wieder gültig ist. Die Anpassung und nicht komplette Neuplanung führt dazu, dass weniger Unruhe innerhalbe der Fertigung entsteht. Lokale Suchverfahren können schon durch geringe Umplanungen, wie das Vertauschen von zwei Operationen, oder durch geringe Verschiebung der Operationen, den Ablaufplan wieder zulässig und durchführbar machen [105]. Die Verwendung von aufwendigeren heuristischen, oder metaheuristischen Optimierungsverfahren ermöglicht die Wiederherstellung der Durchführbarkeit des Ablaufplans mit einer höheren Optimierungsqualität. Die Problemstellung muss zuvor auf die aktuelle Situation angepasst werden. Eine bspw. nicht mehr verfügbare Maschine sowie bereits bearbeitete Operationen müssen dabei berücksichtigt werden. Besonders bei Metaheuristiken kann dies zu einem größeren Aufwand führen, da hier auch zusätzliche Datenstrukturen aktualisiert werden müssen.

Ein weiterer Ansatz zur Anpassung des Ablaufplans an unerwartet auftretende Ereignisse ist die Verwendung von Agenten zur selbstorganisierten Steuerung der Fertigung [106]. Agenten repräsentieren hierbei die einzelnen Ressourcen (Maschine, Handarbeitsplätze …), Transportsysteme und Aufträge bzw. Produkte. Diese führen sowohl die Erfassung der Fertigungsdaten als auch die Umplanung durch. Dadurch kann schnell auf Störungen reagiert werden. Die einzelnen Agenten haben allerdings, wie bereits erwähnt, keinen Überblick über das Gesamtsystem, was zu Problemen bei der Aufrechterhaltung des Ablaufplans sowie der Garantie bzgl. der Fertigstellungstermine führt, und auch eine globale Optimierung des Systems unwahrscheinlich macht.

2.2.5 Optimierungsziele

In der Produktion gibt es neben der bereits erläuterten DLZ, eine Vielzahl von Zielfunktionen, nach welchen die Ablaufplanung optimiert werden kann. Häufig werden hierzu einzelne Kennzahlen, sogenannte Key-Performance-Indicators (KPIs), verwendet. Diese sind im VDMA-Einheitsblatt 66412-1:2009-10, *Manufacturing Execution Systems (MES) – Kennzahlen* definiert sowie mit den entsprechenden Formeln zur Berechnung erläutert [107]. Für die Produktionssteuerung relevante KPIs sind in der Normenreihe ISO 22400 *"Key-Performance-Indicators for Manufacturing Operations Management"* definiert [108], [109]. Eine KPI, die durch die Ablaufplanung beeinflusst wird, ist die Gesamtanlageneffektivität (GAE) (engl. Overall Equipment Effectiveness (OEE)). Diese drückt die ungeplanten Verluste einer Anlage quantitativ aus. Die Planbelegungszeit dient als Ausgangsbasis zur Berechnung der GAE. Sie gibt die theoretisch maximale Belegungszeit einer Anlage, unter Berücksichtigung geplanter Stillstände wie Pausen, Feiertagen oder Wartungen, an. Die GAE berechnet sich durch die Anlagenverfügbarkeit (AV), den Leistungsgrad (LG) sowie die Qualitätsrate (QR) wie folgt:

$$GAE = AV \bullet LG \bullet QR$$

Innerhalb der AV erfolgt die Berücksichtigung ungeplanter Stillstände der Anlage. Hierzu zählen bspw. Rüstvorgänge sowie fehlendes Material oder Personal. Die Verfügbarkeit ergibt sich durch die Subtraktion dieser Stillstände von der Planbelegungszeit, wodurch sich die AV wie folgt berechnet:

$$Verfügbarkeit = Planbelegungszeit - Ungeplante\ Stillstände$$
$$Anlagenverfügbarkeit = Verfügbarkeit\ /\ Planbelegungszeit$$

Der LG berücksichtigt zum einen die Zeiten, in denen sich eine Maschine im Leerlauf befindet und zum anderen die Zeiten, in denen die Maschine mit verringerter Geschwindigkeit läuft. Da von der Ablaufplanung die Materialzufuhr der Maschinen abhängt und diese für Leerlauf der Maschine verantwortlich sein kann, hat die Ablaufplanung einen direkten Einfluss auf den LG. Dieser berechnet sich wie folgt:

$$Leistungsgrad = Istleistung[Stück\ /\ Zeit]\ /\ Solllleistung[Stück\ /\ Zeit]$$

Mit der QR werden alle Verluste, welche durch die Fertigung fehlerhafter Produkte entstanden sind, gemessen und berücksichtigt. Die QR wird wie folgt berechnet:

$$Qualitätsrate = Anzahl\ guter\ Teile\ /\ Anzahl\ Defektteile$$

Die Ablaufplanung hat somit lediglich einen direkten Einfluss auf den LG, weshalb dieser stellvertretend für die GAE als Zielfunktion berücksichtigt werden kann.

Die MA bzw. *Equipment Utilization (EU)* beschreibt den Anteil der Zeit, in der eine Fertigungsanlage genutzt wird [110]. Beträgt die Produktionszeit einer Anlage für eine Woche 6 Tage mit jeweils zwei 8 Stunden Schichten (6x2x8=96 Stunden) und läuft diese durchgängig, ergibt sich bei einer wöchentlichen Gesamtstundenzahl von 168 Stunden, eine Auslastung von 57,14 % (96 Stunden / 168 Stunden). Diese Kennzahl wird durch die Ablaufplanung beeinflusst, da die Ablaufplanung bestimmt, wann welches Produkt auf welcher Maschine bearbeitet wird. Die MA ist dem zuvor beschriebenen LG ähnlich.

Oftmals sind mehrere KPIs von Bedeutung und sollen daher mit dem Ziel, eine höhere Gesamtperformance des RS zu erlangen, optimiert werden. Hierbei wird von multikriterieller Optimierung (MKO) gesprochen. Die Optimierung einer KPI kann jedoch zur Reduzierung einer anderen KPI führen. Zur Entscheidungsfindung bei solchen Konflikten muss die Priorisierung der einzelnen Kennzahlen festgelegt werden. In der Produktionssteuerung muss bspw. entschieden werden, ob eine Minimierung der DLZ wichtiger ist als die Erhöhung der GAE, oder umgekehrt. Zwei grundlegende Verfahren zur Festlegung dieser Priorisierung sind die *Methode der gewichteten Summen (Skalarisierung)* und die *lexikografische Optimierung*. Bei der Skalarisierung wird jede Kennzahl innerhalb der Zielfunktion mit einem Gewichtungsfaktor multipliziert. Ein Beispiel hierfür ist die folgende Zielfunktion:

$$\min_{x,y} f(x,y) \rightarrow f(x,y) = 0,4 \bullet x + 0,6 \bullet y$$

x und *y* sind die zu optimierenden, in diesem Fall zu minimierenden Kennzahlen, wobei *x* mit 0,4 schwächer gewertet wird als *y* mit 0,6. Dies bedeutet, dass *y* gegenüber *x* priorisiert wird. Durch eine geeignete Gewichtung kann definiert werden, wie groß die Bedeutung der einzelnen Kenngrößen ist. Die genaue Gewichtung der einzelnen Kennzahlen erfolgt oft willkürlich, was ein Problem bei der Skalarisierung darstellt.

Die lexikografische Optimierung definiert eine Multi-Kriterien-Politik, welche die verschiedenen Kriterien ordnet: Das erste Kriterium wird als das Wichtigste angesehen, jede Verbesserung dieses Kriteriums ist jeden Verlust bei den anderen Kriterien wert. Diese Politik wird für jedes Kriterium der Ordnung fortgeführt [111]. Liegen für das wichtigste Kriterium mehrere Optima vor, wird die zweitwichtigste Kennzahl zur Entscheidungsfindung herangezogen und im nächsten Schritt die Lösung weiter betrachtet, welche bzgl. der zweitwichtigsten Kennzahl das Optima liefert. Dieser Vorgang wird so lange wiederholt, bis jede Kennzahl innerhalb der Zielfunktion betrachtet wurde, oder für die aktuell betrachtete Kennzahl nur ein Optimum vorliegt [112].

Die Dominanzlösung und der optimale Wert einer MKO werden erreicht, wenn keine Kennzahl innerhalb der Zielfunktion weiter verbessert werden kann, ohne dadurch eine andere Kennzahl innerhalb der Zielfunktion zu verschlechtern. Diese Bedingung wird als *Pareto-Optimalität* bezeichnet [113].

2.3 Algorithmen zur Lösung kombinatorischer Optimierungsprobleme

Ein Überblick zu verschiedenen Algorithmen und Verfahren zur Optimierung von Ablaufplanungsproblemen wird in diesem Kapitel gegeben. Diese werden nach ihrer Art der Lösungsfindung organisiert und gruppiert (siehe Abbildung 11). Die vorgestellten Algorithmen können allgemein zur Lösung kombinatorischer Probleme verwendet werden. Sie werden generell in exakte und heuristischen Verfahren untergliedert. Im Rahmen der heuristischen Verfahren erfolgt eine nochmalige Unterteilung in konstruktive, künstliche Intelligenz (KI) und Verbesserungs-Verfahren. Im Folgenden werden einige der aufgeführten Optimierungsalgorithmen erläutert.

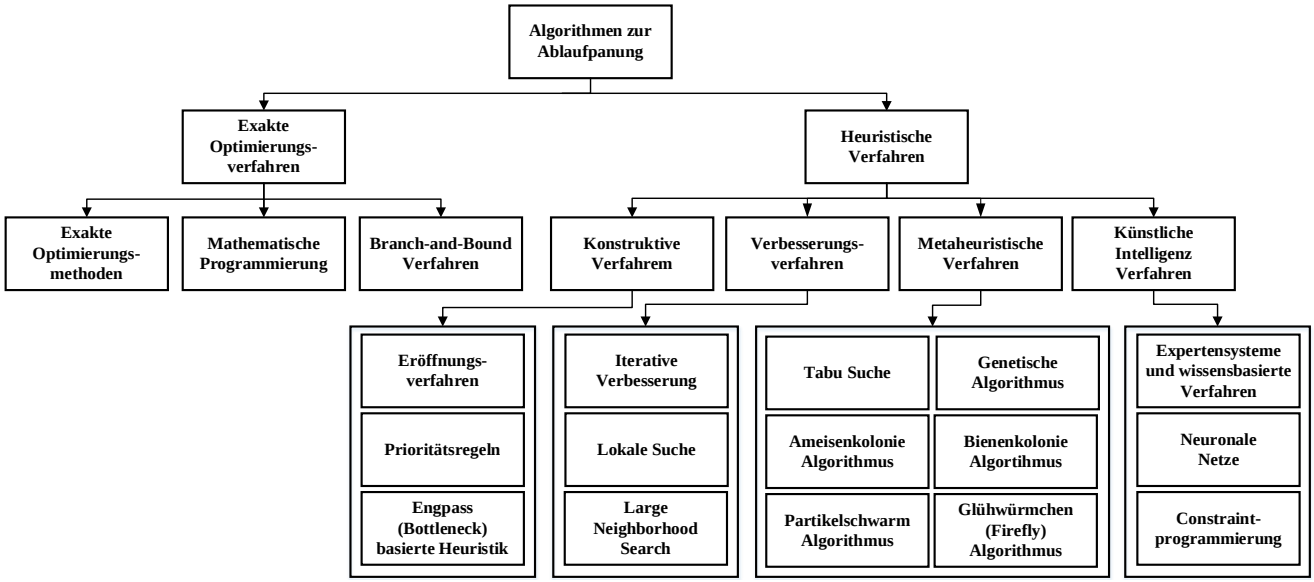


Abbildung 11: Optimierungsalgorithmen zur Ablaufplanung (in Anlehnung an [114])

2.3.1 Exakte Optimierungsverfahren

Exakte Optimierungsverfahren suchen für das vorliegende Problem die bestmögliche Lösung, welche als globales Optimum bezeichnet wird. Ein Methode, die optimale Lösung eines Ablaufplanungsproblems zu finden, ist es, alle möglichen Ablaufpläne zu untersuchen [115]. Dies ist selbst bei kleinen Problemen mit wenigen Aufträgen, Operationen und Maschinen aufgrund der kombinatorischen Explosion rechenzeit-technisch nicht durchführbar. Um dennoch das globale Optimum zu finden, werden Methoden der gemischtganzzahligen (nicht)linearen Programmierung (Mixed Integer Linear Programming, kurz: MILP; Mixed Integer Programming, kurz: MIP) verwendet [116]. Eine beispielhafte Formulierung des JSSP als MIP-Modell ist im Anhang H zu finden.

Bei der Verwendung rein mathematischer Optimierungsverfahren erfolgt die Repräsentation des vorliegenden Ablaufplanungsproblems durch eine Menge von Ungleichungen [117]. Die Lösung des so repräsentierten Problems erfolgt anschließend durch mathematische Verfahren wie bspw. Simplex-Variationen [118], [119]. Die Anzahl von (Un-)Gleichungen, die das vorliegende Problem beschreiben, steigt mit der Anzahl der Restriktionen an das Ablaufplanungsproblem [3].

Das Branch & Bound Verfahren ist ein weiteres Verfahren zur Findung des globalen Optimums eines Ablaufplanungsproblems. Dieses ist bei der Erstellung und Durchführung allerdings deutlich performanter [120], [121]. Branch & Bound führt, basierend auf dem Problem beschreibenden disjunktiven Graph, eine Kombination aus Breiten- und Tiefensuche durch. Als Ergebnis werden (Teil-)Ablaufpläne erstellt. Während der Suche innerhalb des Graphen wird immer der vielversprechendste Zweig verfolgt, wodurch der Lösungsraum effizient erforscht wird. Branch & Bound Verfahren werden in kommerziellen Softwaresystemen oftmals in nächtlichen Planungsläufen eingesetzt [122].

Exakte Optimierungsverfahren können ab einer gewissen Problemgröße den Lösungsraum nicht mehr in vertretbarer RZ durchsuchen. Sie können für das Problem in diesem Fall keine Lösung finden. Gegenüber den anderen aufgezeigten Verfahren ist die benötigte RZ deutlich höher, was auch die in Kapitel 4.3.4 durchgeführte Simulationsstudie zeigt.

2.3.2 Konstruktive Verfahren

Konstruktive Verfahren sind einfache Verfahren, die dazu dienen, einen gültigen Fertigungsplan zu erzeugen. Sie werden oft als Bestandteil sogenannter Eröffnungsverfahren verwendet. Diese führen eine Fertigungssimulation, zur Erstellung eines gültigen Ablaufplans, durch [93], [117]. Die Entscheidungsfindung innerhalb dieser Fertigungssimulation erfolgt unter Verwendung von Prioritätsregeln. Der Ablauf dabei ist folgender:

1. Erstellung einer Warteschlange auszuführender Operationen für jede Maschine
2. Aufnahme aller Operationen ohne Vorgänger in die Warteschlangen
3. Selektion einer Warteschlange
4. Selektion einer Operation aus der in Schritt 3 ausgewählten Warteschlange
5. Aufnahme der Nachfolgeoperation der in Schritt 4 selektierten Operation in die zugehörige Warteschlange

Dieses Vorgehen stellt sicher, dass die Reihenfolgerestriktionen innerhalb eines Auftrags eingehalten werden. Der Prozess wird solange wiederholt, bis alle Warteschlangen leer sind. Als Ergebnis liegt anschließend ein gültiger Ablaufplan vor. Eröffnungsverfahren erstellen die erste Lösung eines Problems, welche später, bspw. durch metaheuristische Verfahren, iterativ verbessert wird.

Für die Auswahl der Warteschlangen und Operationen im zuvor beschriebenen Ablauf gibt es verschiedene Algorithmen. Der Giffler-Thompson Algorithmus (GTA) ist ein hierfür häufig genutzter Algorithmus, da er nachweislich gültige Fertigungspläne erzeugt [123]. Ein beispielhafter Ablauf des GTA ist in Abbildung 12 und Abbildung 13 dargestellt. Hierbei liegt bereits ein teilweise konstruierter Fertigungsplan vor, zu welchem die sich in den Warteschlangen befindlichen Operationen hinzugefügt werden sollen. In den Warteschlangen der einzelnen Maschinen *(M1,…,M3)* befinden sich alle Operationen ohne Vorgänger sowie alle Operationen mit bereits eingeplanten Vorgängern. Im ersten Schritt werden für alle sich in den Warteschlangen befindlichen Operationen die möglichen Endzeitpunkte c_{ji} berechnet. *j* gibt dabei den Auftrag und *i* die Stelle der Operation innerhalb der Operationsreihenfolge des Auftrags an. Der GTA wählt anschließend die Warteschlange aus, in der sich die Operation mit dem frühsten Endzeitpunkt befindet. Im vorliegenden Beispiel ist dies die Warteschlange der Maschine *M2* (1).

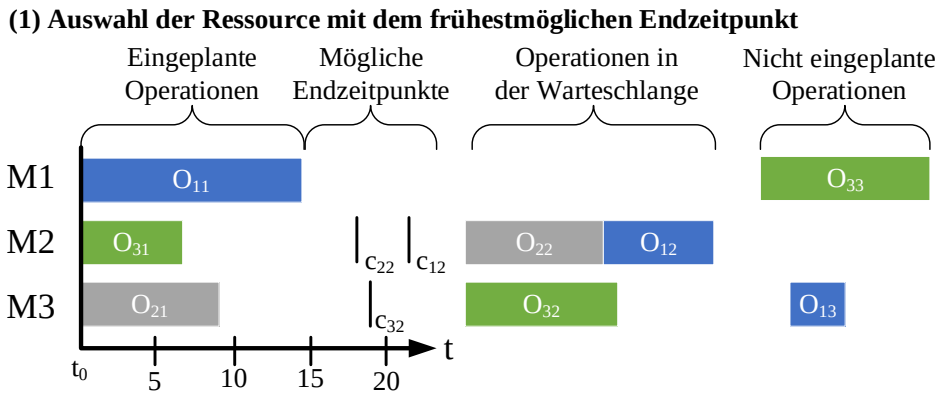


Abbildung 12: GTA: (1) Auswahl der Maschine mit dem frühestmöglichen Endzeitpunkt (nach [3])

Anschließend werden alle Operationen zur Einplanung ausgewählt, deren möglicher Startzeitpunkt s_{ji} vor dem Endzeitpunkt der Operation mit dem frühsten Endzeitpunkt innerhalb der Warteschlange liegen (2). Nachdem die ausgewählten Operationen in den Fertigungsplan eingefügt wurden, werden die

jeweiligen Nachfolger in der entsprechenden Warteschlange aufgenommen. Dieser Ablauf wird solange wiederholt, bis alle Warteschlangen leer sind und ein gültiger Fertigungsplan vorliegt.

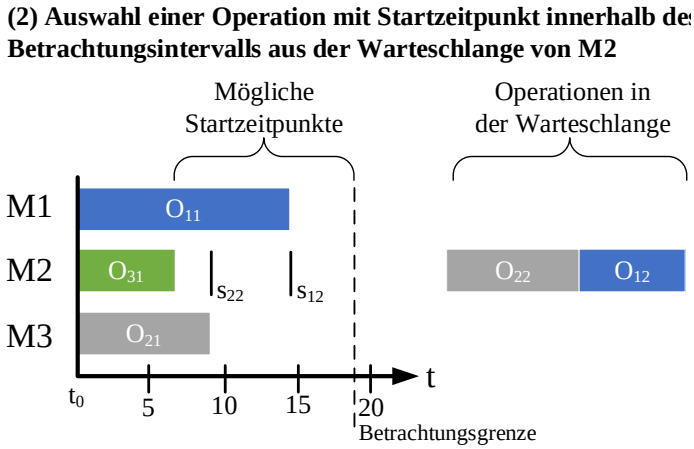


Abbildung 13: GTA: (2) Auswahl einer Operation mit Startzeitpunkt innerhalb des Betrachtungsintervalls aus der Warteschlange von M2 (nach [3])

Greedy oder myopische Heuristiken sind Beispiele für weitere Verfahren zur Auswahl der nächsten Operation innerhalb einer Warteschlange. Diese betrachten die Priorität einer Operation unter der lokalen Situation [3]. Sie nutzen hierzu Prioritätsregeln wie bspw.:

- Kürzeste Bearbeitungsdauer der Operation (shortest process time, SPT).
- Längste Bearbeitungsdauer der Operation (longest process time, LPT).
- Frühester Fertigstellungszeitpunkt des Auftrags der Operation (earliest due date, EDD)

Neben diesen einfachen Prioritätsregeln gibt es auch Prioritätsregeln, die weiterführende Betrachtungen in ihre Auswahlentscheidung mit einbeziehen. Hierzu zählt die Berücksichtigung der verbleibenden Bearbeitungsdauer des Auftrags der Operation (remaining time, RT), die Anzahl an Operationen innerhalb der Folgewarteschlangen (work in next queue, WINQ), zufällige Auswahl (RANDOM) usw. In der Literatur wurden bereits über hundert Prioritätsregeln vorgestellt und untersucht, wobei sich die Prioritätsregeln SPT und RANDOM trotz ihrer Einfachheit immer wieder durch sehr gute Ablaufpläne hervorgetan haben [124]. Die Verwendung von Prioritätsregeln beschränkt sich nicht auf zentrale Ablaufplanungsverfahren, sondern ist auch bei dezentralen Steuerungsansätzen vorzufinden. Eröffnungsverfahren haben den Nachteil, dass sie lediglich die anstehende Auslastung innerhalb der Warteschlange betrachten und so keine Engpassmaschinen identifizieren können. Um dieses Problem mit zu berücksichtigen, dient das Shifting-Bottleneck Verfahren (Engpassverfahren), welches allerdings nicht genauer erläutert wird. Weitere Konstruktionsverfahren, sind die Erstellung mehrerer Ablaufpläne mittels Lagrange Relaxation [125] oder mittels einer Kombination aus mehreren Prioritätsregeln [126]. Aus den erzeugten Ablaufplänen wird der Beste ausgewählt. Die Vorteile der heuristischen Konstruktionsverfahren liegen in ihrer einfachen Implementierung sowie der geringen RZ. Aus diesen beiden Gründen haben heuristische Konstruktionsverfahren in kommerziell erhältlichen System zur Produktionssteuerung einen hohen Verbreitungsgrad [3].

2.3.3 Verbesserungsverfahren

Heuristische Verbesserungsverfahren nutzen zuvor erzeugte Ablaufpläne, um basierend auf diesen, iterativ Verbesserungen durchzuführen. Ein durch Konstruktionsverfahren erzeugter Ablaufplan dient hierbei oft als Ausgangsbasis. Ein heuristisches Verbesserungsverfahren ist die Lokale Suche. Diese

betrachtet die benachbarten Ablaufpläne des aktuellen Ablaufplans, mit dem Ziel eine bessere Lösung zu finden [127], [128], [129]. Sie untersucht, anders als exakte Verfahren, nicht den gesamten Lösungsraum und hat somit nicht den Anspruch, das globale Optimum zu finden. Die Erzeugung von neuen, benachbarten Ablaufplänen erfolgt mittels einfacher Modifikationsregeln. Von den so erhaltenen Ablaufplänen werden solche weiter untersucht, die eine bessere Lösung bzgl. des Zielfunktionswerts liefern als der aktuelle Ablaufplan. Lokale Suchverfahren optimieren nur einen Teil des Ablaufplans, wodurch diese Verfahren in ein lokales Optimum geraten können. Einfache, lokale Suchverfahren finden deshalb selten alleinstehend Verwendung, sondern bilden oft einen Teilprozessschritt im Rahmen komplexer Optimierungsstrategien. Ein gültiger Ablaufplan liegt in der Regel nach jedem Iterationsschritt vor, wodurch diese Verfahren jederzeit unterbrechbar sind. Sie eignen sich deshalb besonders für die permanente Ausführung [3].

Die *Large Neighborhood Search (LNS)* stellt eine Weiterentwicklung der lokalen Suche dar. Die Grundidee der LNS besteht darin, dass eine große Nachbarschaft es ermöglicht, sich leicht im Lösungsraum zu bewegen, auch wenn das Problem durch Randbedingung stark eingeschränkt ist. Auf kleinen Nachbarschaften basierende Ansätze haben bei der Durchsuchung des Lösungsraums hingegen wesentlich mehr Schwierigkeiten [130]. Eine explizite Trennung zwischen kleiner und großer Nachbarschaft anhand der Anzahl an Nachbarn ist nicht zu geben. Pisinger und Ropke [130] definieren eine Nachbarschaft als groß, wenn diese exponentiell mit der Größe der Instanz wächst, oder wenn die Nachbarschaft zu groß ist, um in der Praxis explizit durchsucht werden zu können. Der genaue Ablauf der LNS wird folgend erläutert. Als Beispiel einer initialen Lösung dient der in Abbildung 14 gezeigt Ablaufplan. Dieser basiert auf dem JSSP-Beispiel in Kapitel 2.2.1. Während jeder Iteration werden einzelne Fragmente der Lösung (Operationen) innerhalb einer definierten Nachbarschaft miteinander vertauscht, um neue Lösungen zu erzeugen. Die Nachbarschaft *M(x)* einer Lösung *x* entspricht der Menge an Lösungen, die sich durch das Vertauschen von Fragmenten ergibt. Sie schließt ausschließlich gültige Lösungen ein. Die beiden Operationen O_{12} und O_{22}, werden in Abbildung 14 als Nachbarschaft *N(x)* definiert. Die Auswahl der zu vertauschenden Fragmente erfolgt bei der LNS in der Regel zufällig, sodass bei jedem Iterationsschritt unterschiedliche Fragmente der Lösung vertauscht werden. Der Nachbarschaft *N(x)* in Abbildung 15 gehören zwei Elemente an, sodass diese folglich miteinander vertauscht werden.

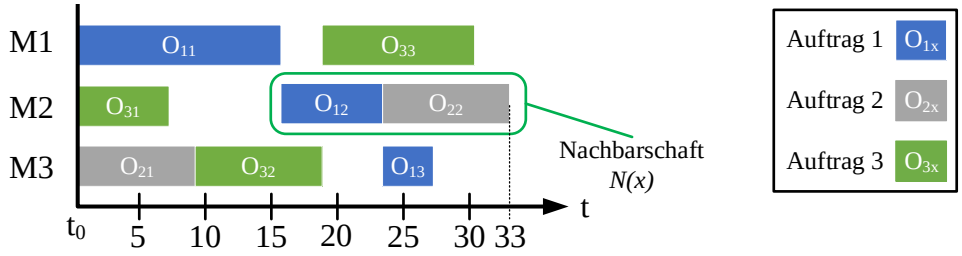


Abbildung 14: Initiale Lösung und Definition einer Nachbarschaft *N(x)* bei der LNS.

Das Vertauschen der Operationen kann eine Verletzung geltender Reihenfolge- oder Ressourcenbedingungen hervorrufen und den Ablaufplan dadurch ungültig machen. Dies ist auch in Abbildung 15 der Fall, wo sich die Operationen O_{12} und O_{13} des ersten Auftrags zeitlich überschneiden. Eine Verletzung der geltenden Reihenfolge-Randbedingung ist die Folge.

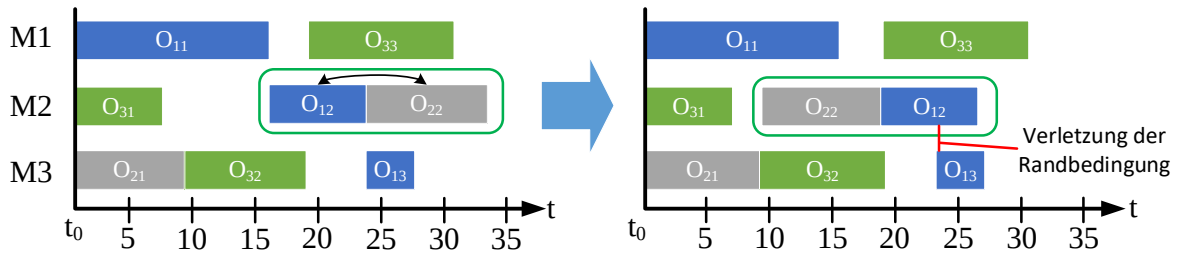


Abbildung 15: Vertauschung der Elemente in der Nachbarschaft *N(x)* und sich ergebende Verletzung der Randbedingungen.

Die Reparatur der neuerzeugten Lösung erfolgt deshalb als nächster Schritt. Die Lösung wird soweit verändert, bis alle Randbedingungen wieder eingehalten werden. Für das vorliegende Beispiel bedeutet dies, dass der Startzeitpunkt von Operation O_{13} so verschoben werden muss, dass er nach dem Endzeitpunkt seiner Vorgängeroperation O_{12} liegt (Abbildung 16). Der so erzeugte und wieder gültige Ablaufplan besitzt eine geringe DLZ als die initiale Lösung. Im folgenden Iterationsschritten dient die neue Lösung somit als Ausgangsbasis.

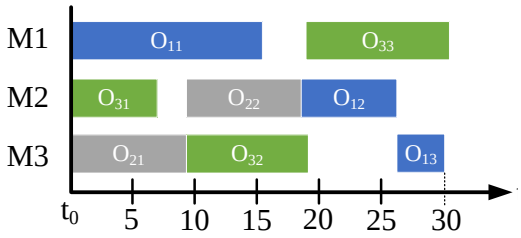


Abbildung 16: Neuer, gültiger Ablaufplan mit reduzierter DLZ gegenüber der initialen Lösung in Abbildung 14.

2.3.4 Metaheuristische Verfahren

Zur Lösung kombinatorischer Probleme gibt es einige, von dem konkreten Problem unabhängige, Lösungsstrategien. Bei diesen Verfahren handelt es sich um iterative Verbesserungsverfahren, welche auf vordefinierten Schemata basieren. Die Verbesserung startet häufig, wie bei lokalen Suchverfahren, basierend auf einem initialen, durch Konstruktionsverfahren erzeugten Ablaufplan. Diese Verfahren werden als Metaheuristiken bezeichnet und können in zwei Kategorien untergliedert werden:

- Einzellösungsbasierte Metaheuristiken
- Populationsbasierte Metaheuristiken

Sowohl bei einzellösungs- als auch bei populationsbasierten Metaheuristiken sind oft natur-inspirierte Algorithmen zu finden, wie bspw. die Ameisenoptimierung oder die Partikelschwarmoptimierung. Aus der Natur bekannte Verhaltensweisen werden in programmiertechnische Lösungen überführt, um den Lösungsraum kombinatorischer Probleme effizient zu durchsuchen [131].

Einzellösungsbasierte Metaheuristiken sind häufig dafür ausgelegt, die zuvor beschriebenen, einfachen iterativen Verbesserungsverfahren, wie lokale Suche (LS), zu steuern. Ein Beispiel für eine solche Metaheuristik ist die von Glover [132] entwickelte Tabu-Suche (TS). Die TS erzeugt im ersten Schritt eine initiale Lösung und wendet auf diese anschließend ein lokales Suchverfahren an. Die erhaltenen Lösungen werden in einer Tabu-Liste gespeichert. Die beste Lösung, welche noch nicht in der Tabu-Liste vorhanden ist, wird vom Algorithmus als Ausgangslösung für die nächste iterative Durchführung der lokalen Suche verwendet. Dadurch soll das erneute Untersuchen bereits betrachteter Lösungen

verhindert werden [133]. Dieser Ablauf wird solange wiederholt, bis die TS eine definierte Abbruchbedingung erfüllt. Neben der klassischen TS gibt es auch viele modifizierte und weiterentwickelte Varianten dieser. Der TS-Algorithmus ist zur Verdeutlichung folgend als Pseudo-Code dargestellt:

```
1. Erstellung einer initialen Lösung
2. Beste Lösung = Initiale Lösung
3. Beste Lösung zur Tabu-Liste hinzufügen
4. Solang (Abbruchbedingung nicht erfüllt)
5.   Erstelle Nachbarschaft durch lokale Suche basierend auf bester Lösung
6.   Auswahl der besten Lösung innerhalb der Nachbarschaft
7.   Solange (Beste Lösung in Tabu-Liste vorhanden)
8.           Beste Lösung = Nächstbeste Lösung aus Tabu-Liste
9.   Ende Solange
10.  Alle neuen Lösungen zur Tabu-Liste hinzufügen
11.  Ende Solange
```

Abbildung 17: Pseudo-Code zum Ablauf der Tabu-Suche.

Die Vorteile von einzellösungsbasierten Metaheuristiken sind die einfache Implementierung und der Umstand, dass sie zu jedem Zeitpunkt abgebrochen werden können und dennoch einen zulässigen Fertigungsplan liefern. Zudem ermöglichen sie eine deutliche Verbesserung der durch Prioritätsregelverfahren erzeugten Ablaufpläne.

Eine der bekanntesten und ältesten populationsbasierten Metaheuristiken ist der genetische Algorithmus (GA). Dieser findet in vielen Auflaufplanungsproblemen Anwendung [134], [135]. Das allgemeine Ablaufschema eines GA ist in Abbildung 18 dargestellt. Der GA startet mit der einmaligen Initialisierung der Population, welche aus einer Menge von Individuen besteht. Jedes Individuum stellt eine Lösung, bspw. einen Ablaufplan, dar. Im nächsten Schritt wird jedes Individuum evaluiert und erhält in diesem Schritt eine Bewertung (fitness). In der Ablaufplanung ist dies bspw. die DLZ. Anschließend wird im Schritt Selektion entschieden, welche Individuen der neuen und alten Generation überleben und somit die Population für die nächste Iteration bilden. Bei der nachfolgenden Mutation entstehen durch die Kreuzung von zwei Individuen (parents) neue Individuen (siblings). Diese stellen die neue Generation dar. In der nachfolgenden Mutation werden alle Individuen der neuen Generation verändert. Die beispielhafte Anwendung eines GA auf das JSSP ist im Anhang I zu finden.

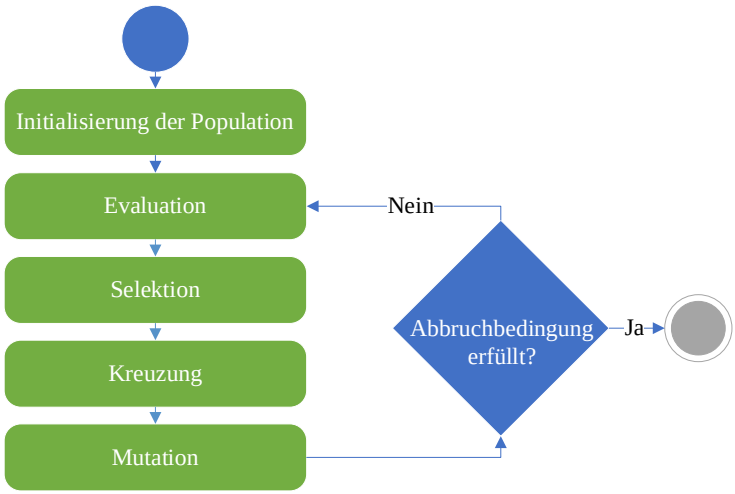


Abbildung 18: Ablaufschema GA

Eine weitere Art von populationsbasierten Metaheuristiken sind die sogenannten Ameisenalgorithmen wie Ant Colony Optimization (ACO), oder Ant System (AS). Diese Verfahren wurden vor ca. 25 Jahren erstmals auf die Problemstellung der Shop Scheduling Probleme adaptiert [136]. Basis der Ameisenalgorithmen ist die Nachbildung des Verhaltens von Ameisen bei der Nahrungssuche. Ameisen versprühen bei der Suche nach Nahrung eine Pheromonenspur auf dem von Ihnen zurückgelegten Weg. Über die Zeit nimmt die Pheromonenkonzentration (PK) ab und Ameisen folgen immer nur dem Weg mit der höchsten PK. Gehen zwei Ameisen einen unterschiedlich langen Weg zur Nahrungsquelle, wird bei der Rückkehr der Ameisen die PK auf dem kürzeren Weg höher sein, da die entsprechende Reisezeit der Ameise geringer war und so weniger Pheromon „verdampfen“ konnte. Die nächste Ameise wird entsprechend diesem Weg folgen, was die PK auf diesem Weg weiter erhöht, sodass auf dem kürzesten Weg zur Nahrungsquelle immer die höchste PK vorliegt. Bei der Anwendung des Ameisenalgorithmus auf Shop Scheduling Probleme durchläuft mindestens eine „Ameise“ alle Knoten des problembeschreibenden disjunktiven Graphs und erstellt dadurch einen Ablaufplan [137]. Neben den vorgestellten populationsbasierten Metaheuristiken gibt es noch eine Vielzahl weiterer solcher Verfahren. Beispiele hierfür sind: Künstliche Bienenkolonie [138], Partikelschwarmintelligenz [139] oder Glühwürmchen Algorithmus [140]. Diese werden der Übersichtlichkeit halber nicht genauer erläutert, sondern auf die angegebenen Literaturstellen verwiesen.

2.3.5 Künstliche Intelligenz Verfahren

Auch aus dem Bereich der KI gibt es Verfahren, die im Rahmen der Ablaufplanung eingesetzt werden können [141].

Künstliche Neuronale Netzwerke (KNN) versuchen die Funktionalität biologischer, neuronaler Netzwerke nachzubilden und bestehen aus einem Netz von künstlichen Neuronen. Vorteil bei der Verwendung eines KNN ist dessen Fähigkeit zu Lernen, weshalb sich diese Technologie dazu eignet, in großen Datensätzen Muster zu erlernen und anschließend entsprechende Auffälligkeiten oder erlernte Muster zu erkennen. Anwendungsbeispiele hierfür sind die Spracherkennung und Bildverarbeitung. Nachteilig ist, dass für den Prozess des Erlernens große Datensätze vorhanden sein müssen. Im Rahmen der Produktionssteuerung eignet sich die Anwendung von KNN deshalb weniger in der Erstellung des Ablaufplans bzw. der direkten Steuerung des Systems als vielmehr in unterstützenden Tätigkeiten. Die vorzeitige Erkennung von notwendigen Wartungen an den einzelnen Ressourcen kann bspw. durch die Verwendung eines KNN erfolgen und in der Ablaufplanung entsprechend prädiktiv mitberücksichtigt werden. Eine Möglichkeit maschinelles Lernen dennoch für die Lösung von JSSP einsetzen zu können, ist die Verwendung des Lernverfahrens Reinforcement Learning (RL). Dieses Verfahren benötigt keine großen Datensätze, sondern lernt kontinuierlich während der Ausführung. Hierfür wird eine Aktion durchgeführt, das Ergebnis dieser bewertet und die Bewertung anschließend also Belohnung oder Bestrafung in das KNN in Form von Gewichtungen zurückgeführt. Dadurch werden zum Erfolg führende Aktionen positiv, und zum Misserfolg führende, entsprechend negativ gewichtet und entsprechend erlernt. Liu et al. [141] nutzen für die Lösung und Optimierung eines JSPP eine Kombination aus KNN und Reinforcement Learning (RL). Der Ansatz wird mit Prioritätsregeln und einem metaheuristischen Verfahren verglichen. Er liefert schlechtere Ergebnisse bzgl. der DLZ-Minimierung als das zum Vergleich herangezogene metaheuristische Verfahren von Huang und Liao [142]. Weiteres Problem bei der Nutzung von RL ist, dass das KNN während des laufenden Betriebs beginnt zu lernen und somit einige Zeit benötigt, um sich an das vorliegende Problem anzupassen.

Expertensysteme beinhalten eine Wissensbasis mit Fallbedingungen (Wenn A, dann B), welche den Menschen bei der Lösungsfindung innerhalb komplexer Probleme unterstützen soll. Problem bei der Verwendung von Expertensystemen ist, dass zur Lösung spezieller Probleme eine große Wissensbasis notwendig ist. Dies ist mit erheblichem Aufwand verbunden, denn diese Wissensbasis muss von einem Experten auf dem entsprechenden Gebiet erstellt und gepflegt werden. Eine Methodik zur Nutzung eines solchen Systems ist es, basierend auf der Wissensbasis, einen Entscheidungsbaum zu erstellen. Das System soll innerhalb des Entscheidungsbaums den effizientesten Pfad finden, welcher vom vorliegenden Anfangs- zu dem gewünschten Endzustand führt. Die einzelnen Stationen innerhalb dieses Pfads entsprechen der notwendigen Aktionsreihenfolge. Diese Methodik eignet sich zur Anwendung im Rahmen der Produktionsplanung, bspw. um festzulegen, welche Prozessschritte, in welcher Reihenfolge notwendig sind, um ein Produkt herzustellen bzw. dieses einer Refabrikation zu unterziehen.

Das KI-Verfahren CP eignet sich für die Erstellung und Optimierung von Ablaufplänen. Die generelle Aufgabe der CP ist es, bzgl. eines durch Constraints beschriebenen Problems, aus einer Anzahl von Möglichkeiten eine gültige Lösung zu finden. Man spricht hier von einem *Constraint Satisfaction Problem* (CSP). Bei der Optimierung besteht das Ziel hingegen darin, eine möglichst optimale Lösung zu finden. Auch wenn die Definition einer Zielfunktion bei der CP nicht erforderlich ist, ist es dennoch möglich, eine Zielfunktion vorzugeben und bzgl. dieser nach einer optimalen Lösung zu suchen. Im einfachsten Fall erfolgt ein Vergleich des Zielwerts aller gültigen, gefundenen Lösungen. In diesem Fall spricht man von einem *Constraint Optimization Problem* (COP). Das Problem wird im Rahmen der CP durch unterschiedliche Constraints (Rand- bzw. Nebenbedingungen) definiert. Ein Constraint kann als eine mathematisch formulierte Beziehung (Relation) zwischen Variablen, welche Werte aus ihrem jeweiligen Definitionsbereich (Domäne) annehmen, verstanden werden. Constraints eignen sich somit für die deklarative Modellierung von Problemen. Die Lösungen eines durch Constraints beschriebenen Problems sind die Variablenbelegungen, welche die Konjunktion aller verwendeten Constraints einhalten [143]. Das Berechnungsmodell zur Lösung eines COP besteht aus der Kooperation zweier Komponenten [143]:

- *Constraint Löser*: Reduziert die Domäne der Variablen
- *Suchmechanismus*: Zerlegung des Problems in Teilprobleme

Die Funktionsweise eines Constraint-Lösers wird aus Gründen der Übersichtlichkeit im Anhang J erläutert. Für den Suchmechanismus werden häufig lokale Such- bzw. Verbesserungsverfahren verwendet. Diese führen ihre Optimierungsmechanismen auf einer mittels initialer Variablenbelegung erzeugten Lösung durch.

Die Vorteile bei der CP bestehen darin, dass bei einer Änderung der Nebenbedingungen untersucht wird, welche bereits vorhandenen Lösungen dennoch gültig bleiben, wodurch sich CP auch zur Lösung großer, reeller Ablaufprobleme eignet [144]. Zudem ist die Anpassung des CP-Problems an geänderte Randbedingungen des realen Problems schnell und einfach durchführbar.

Nachdem in diesem Kapitel unterschiedliche Verfahren zur Lösung und Optimierung von Ablaufplanungsproblemen vorgestellt wurden, wird im folgenden Kapitel der Einsatz FTS zur Realisierung eines flexiblen Materialtransports erläutert. Dabei wird besonders auf die Integration eines flexiblen Materialtransports in die Produktionssteuerung sowie auf die SAMF eingegangen.

2.4 Intralogistik mit fahrerlosen Transportsystemen

Ein Problem bei der wirtschaftlichen Umsetzung des Refabrikation ist die Intralogistik [145]. Diese muss sowohl bei der Refabrikation, als auch bei variantenreicher und kundenindividueller Fertigung flexible gestaltet werden. Der Begriff Intralogistik wird durch den Verband Deutscher Maschinen- und Anlagenbau (VDMA) wie folgt definiert: „*Die Intralogistik umfasst die Organisation, Steuerung, Durchführung und Optimierung des innerbetrieblichen Materialflusses, der Informationsströme sowie des Warenumschlags in Industrie, Handel und öffentlichen Einrichtungen.*“ [146]. In diesem Kapitel erfolgt eine Einführung in die Grundlagen von FTS sowie die Darstellung deren Integration in die Produktionssteuerung. Des Weiteren wird der Ansatz der SAMF vorgestellt.

FTS planen ihren genauen Pfad zwischen zwei Punkten, unter der Verwendung von Pfadplanungsalgorithmen, selbstständig. Sie nutzen hierzu eine 2D Karte ihrer Umgebung und können sich innerhalb dieser eigenständig lokalisieren. Das FTS kann durch an ihm angebrachte Sensorik seine aktuelle Umgebung wahrnehmen, wodurch unerwartet auftretende Hindernisse erkannt werden können. Das FTS kann daraufhin eine Pfadneuplanung unter der Einbeziehung der aktuellen Umwelt durchführen und das Hindernis anschließend umfahren. Dadurch besitzt es einen gewissen Grad an Autonomie. FTS sind mit einer Leitsteuerung vernetzt, welche die Aufgabenverwaltung übernimmt. FTS werden im industriellen Bereich für unterschiedlichste Einsatzszenarien genutzt. Die transportierten Güter können von kleinen Transportboxen bis zu tonnenschweren Flugzeugteilen reichen. Durch das breite Einsatzspektrum ergeben sich unterschiedlichste Größen bzw. Traglasten sowie Formen. Ein Beispiel ist das in Abbildung 19 zu sehende FTS *MiR100* der Firma *Mobile Industrial Robot*. FTS sind zusätzlich zum industriellen Umfeld bspw. auch in Krankenhäusern für den Transport von Lebensmitteln zum Patienten im Einsatz. Die ersten FTS waren bereits 1954 in Industrieanlagen im Einsatz [149].

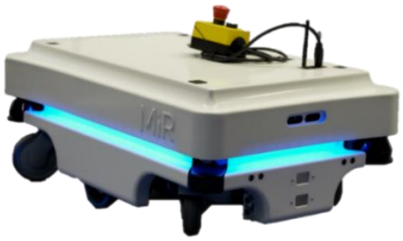


Abbildung 19: FTS *MiR 100* der Firma *Mobile Industrial Robots*.

Die Steuerung eines FTS kann in fünf grundlegende Steuerungsaufgaben unterteilt werden. Diese sind in Abbildung 20 aufgeführt und bestehen aus den Teilaufgaben Aufgabenzuteilung, Lokalisierung, Pfadplanung, Bewegungsplanung und Fahrzeugmanagement [147]. Die Aufgabenzuteilung erstellt einen gültigen Ablaufplan, bei dem die durchzuführenden Transportfahrten den vorhanden FTS zugeordnet werden und entsprechende Reihenfolgebedingungen definiert werden. Eine detailliertere Erläuterung der Aufgabenzuteilung erfolgt in Kapitel 2.4.2. Basierend auf dem Ergebnis der Aufgabenzuteilung muss der genaue Pfad, auf welchem das FTS zwischen zwei Positionen A und B verfährt, ermittelt werden. Ziel der Pfadplanung ist es, einen kollisionsfreien sowie möglichst kurzen Pfad zwischen den beiden Positionen zu ermitteln. Die aktuelle Position des FTS muss hierfür vorab ermittelt werden. Verschiedene Verfahren und Algorithmen sind zur Lokalisierung des FTS innerhalb seiner Umgebung vorhanden. Auf diese Verfahren wird allerdings nicht genauer eingegangen. Da sich das FTS in einer nicht deterministischen Umgebung befindet, ist die tatsächliche Durchführbarkeit des geplanten Pfades nicht garantiert. Ein Objekt oder eine Person kann sich bpsw. auf den geplanten Pfad begeben, wodurch das Weiterverfolgen dieses nicht mehr möglich ist. Die nächste Aufgabe der FTS-

Steuerung besteht deshalb darin, Echtzeit-Anpassungen an dem geplanten Pfad vorzunehmen. Das FTS erfasst hierzu kontinuierlich durch entsprechende Sensorik seine Umgebung, und kann Objekte und Personen auf dem geplanten Pfad detektieren. Wird ein Objekt detektiert, führt das FTS eine erneute Pfadplanung unter Berücksichtigung der aktuellen Umgebung durch, und ist in der Lage, das Objekt zu umfahren. Ist dies nicht möglich, weil eine Umfahrung des Objektes zwangsläufig zu einer Kollision mit anderen Objekten führen würde, wird die fünfte Steuerungsaufgabe, das Fahrzeugmanagement, aktiv. Im Rahmen dieser Steuerungsaufgabe wird im vorliegenden Fall bspw. eine entsprechende Fehlermeldung an einen Werker gegeben. Dieser kann im Anschluss das Problem begutachten und, wenn möglich, dieses durch Entfernen des, den Pfad blockierenden, Objektes beheben. Weitere Aufgaben des Fahrzeugmanagements bestehen in der Überwachung des Batteriezustands, der Einleitung von Ladevorgängen sowie die Informierung über notwendige Wartungsmaßnahmen am FTS.

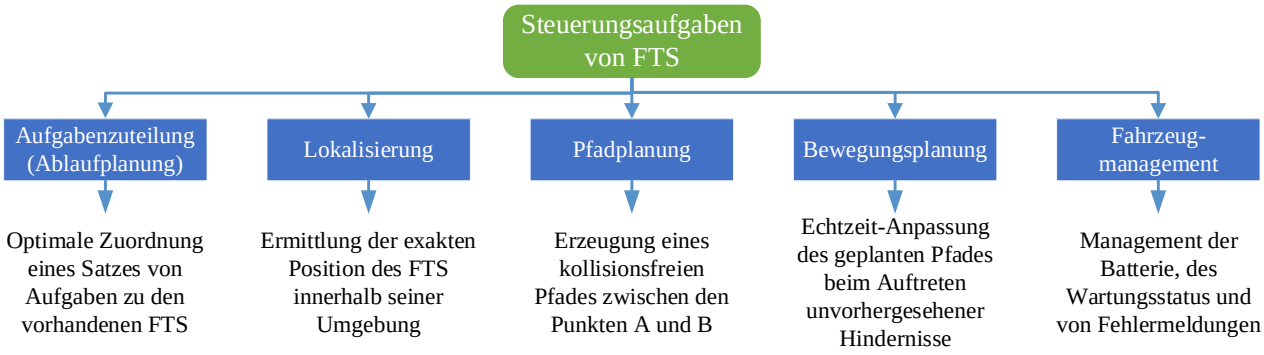


Abbildung 20: Überblick über die fünf Hauptaufgaben zur Steuerung von FTS nach [147].

2.4.1 Flottenmanagement-Systeme

In den meisten industriellen Anwendungen werden FTS durch Flottenmanagement-Systeme in die Produktionssteuerung integriert. Diese sind mit dem jeweils vorhandenen Produktionssteuerungssystem (bspw. MES) verbunden. Eine solche Integration ist konzeptionell in Abbildung 21 dargestellt. Das Flottenmanagement-System erhält aus dem MES die auszuführenden Transportaufträge. Anschließend ordnet es den verfügbaren FTS die durchzuführenden Transportaufträge im Rahmen der Aufgabenzuteilung zu. Hierbei kann auch die Berücksichtigung anderer Randbedingungen wie Batterieladezustände, Eignung des FTS für den jeweiligen Transport, oder mögliche Kollisionen zwischen den geplanten Pfaden mehrerer FTS erfolgen. Eine reaktive Integration von FTS in die Produktionssteuerung, also ohne die Ankopplung an ein MES, ist ebenfalls häufig vorzufinden. Die einzelnen Maschinen fordern hier, bspw. bei der Fertigstellung eines Produktes, selbstständig ein FTS an, welches das Produkt anschließend an der Maschine aufnimmt und zu einer fest definierten Endposition weitertransportiert. Durch den Wegfall einer zentralen Planungseinheit ist bei dieser Methodik keine globale Optimierung möglich.

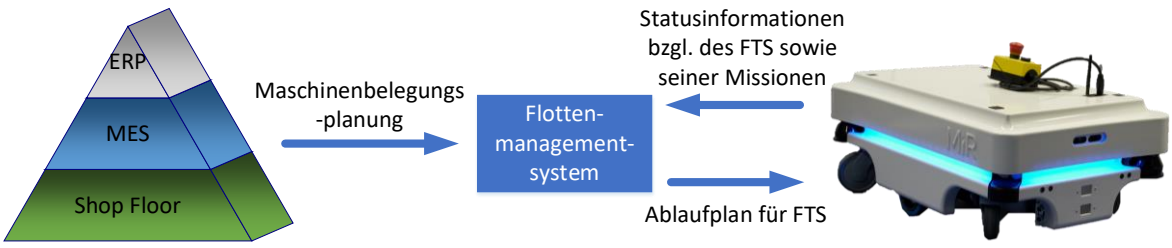


Abbildung 21: Sequenzielle Ablaufplanung unter der Verwendung von MES und Flottenmanagement-System.

Verschiedene Flottenmanagement-Systeme sind kommerziell erhältlich und werden oftmals von den Herstellern der FTS angeboten. Mobile Industrial Robots bietet bspw. die *MiRFleet* [148] an, welche die kollisionsfreie Pfadplanung mehrerer FTS ermöglicht. Das System bietet zudem die Möglichkeit Transportaufträge mit Prioritätsregeln zu versehen. Es überwacht darüber hinaus den Ladezustand der Batterien der FTS und steuert automatisch die notwendigen Ladevorgänge. Die *KUKA.NavigationSolution* der KUKA AG[149] und *der AGV-Manager* von BA Systèmes [150] versuchen die Gesamtfahrzeit unter Berücksichtigung der Fertigungsumgebung, des Verkehrs sowie des gewünschten Zielortes zu reduzieren. Diese Systeme geben Rückmeldung an das übergeordnete Steuerungssystem bzgl. der neuen Ankunftszeiten der einzelnen Aufträge an den entsprechenden Maschinen. Das System *E'tricc* von DEMATIC [151] ordnet den verfügbaren FTS die einzelnen Transportaufträge zu, indem es den Arbeitsablauf analysiert und die Aufträge neu bewertet. Eine Analyse von historischen Reiserouten und Betriebsdaten findet im *AGV MANAGER* von Sidel [152] und im *SGV Manager* von JBT [153], mit dem Ziel die Leistung der FTS-Flotte im industriellen Umfeld zu optimieren, statt. Der *Vehicle Manager* von savant automation [154] ist in der Lage, Inputs von Netzwerkcomputersystemen, diskreten I/Os, SPS-Netzwerken usw. zu verarbeiten, um die verfügbaren FTS den Aufgaben zuzuordnen. Dieser berücksichtigt, im Rahmen seiner Entscheidungsfindung, zudem historische Daten.

Neben den aufgeführten, kommerziell verfügbaren Softwarelösungen ist das Thema Flottenmanagement auch Gegenstand der Forschung. Srivastava et al. [155] stellen z. B. einen agentenbasierten Ansatz zur Steuerung einer FTS-Flotte vor. Ziel ist es, einen kollisionsfreien und zeitoptimierten Pfad innerhalb definierter FTS-Pfadnetze zu finden. Die Validierung des Ansatzes erfolgt anhand verschiedener Simulationsszenarien. Cardarelli et al. [156] präsentieren eine kooperative Cloud-Robotik-Architektur für das Flottenmanagement in industriellen Logistikzentren. Die kooperative Datenfusion aus verschiedenen Sensorsystemen ermöglicht eine ständig aktualisierte globale Live-Ansicht der Umwelt. Dadurch sollen unerwartete Hindernisse in der Umwelt erkannt und kollisionsfrei umfahren werden können. Die erfolgreiche Validierung der Methodik erfolgt in einer realen, industriellen Umgebung. Yao et al. [157] stellen ein *Smart AGV Management System* vor, um die Planung von FTS in einem Fertigungsprozess zu optimieren. Der vorgeschlagene Ansatz nutzt die Kombination aus Echtzeit-Datenanalyse und einem Digital Twin-Modell (digitaler Zwilling) zur Optimierung der Aufgabenverteilung unter den verfügbaren FTS. Der Ansatz kann an einem Demonstrator mit einer manuellen Montagestation erfolgreich validiert werden.

Keines der aufgeführten kommerziellen Systeme und akademischen Konzepte bietet die Möglichkeit zur SAMF. Eine Berücksichtigung von Echtzeit-Fertigungsdaten findet in Flottenmanagement-Systemen ebenfalls oft nicht statt, ist zur Reaktion auf unerwartete Ereignisse jedoch grundlegend notwendig. Zusätzlich stellen alle vorgestellten kommerziellen Flottenmanagement-Systeme eine rein zentrale Steuerungsarchitektur dar und sind dadurch mit den bereits erwähnten Problemen solcher Ansätze behaftet.

2.4.2 Simultane Ablaufplanung von Maschinen und fahrerlosen Transportsystemen

Die SAMF betrachtet die beiden voneinander abhängigen und sich gegenseitige beeinflussenden Ablaufplanungsprobleme der Maschinenbelegung und Transportplanung als ein ganzheitliches Problem. In der Industrie werden diese aktuell getrennt betrachtet, wodurch sich Defizite bei DLZ - und MA – Optimierung ergeben. Eine Begründung hierfür ist, dass die Transportzeiten zu den Stillstandszeiten der Maschinen beitragen, wenn eine Maschine bspw. auf das nächste, zu bearbeitende Produkt warten muss. Dennoch widmen sich nur wenige Arbeiten der SAMF.

In Abbildung 22 ist ein gemeinsamer Ablaufplan für Maschinen *(M1.... M4)* und FTS *(AGV1, AGV2)* in Form eines Gantt-Diagramms dargestellt. Dieses stellt auf der Abszisse die Zeit und auf der Ordinate die vorhandenen Maschinen und FTS dar. Jeder farbige Block stellt entweder eine durchzuführende Operation, oder einen Transportauftrag dar. Die farbigen Blöcke innerhalb der Zeile einer Maschine repräsentiert je eine Operation O_{xy}, die auf dieser Maschine ausgeführt wird. *x* beschreibt, welchem Auftrag die Operation zugehört und *y* an welcher Stelle der Operationsreihenfolge sich die entsprechende Operation befindet. O_{32} ist bspw. die zweite Operation des dritten Auftrags. Innerhalb der Zeile eines FTS stellen blaue Blöcke die beladenen Fahrten und pinke Blöcke die Leerfahrten dar. Das FTS verfährt bei Leerfahrten zwar, transportiert allerdings kein Produkt, wohingegen bei beladenen Fahrten der eigentliche Transport eines Produktes zwischen zwei Maschinen stattfindet. Die Leerfahrten-Blöcke beinhalten auch evtl. Wartezeiten der FTS. Diese können auftreten, wenn ein FTS nach Ankunft an einer Maschine auf das Bearbeitungsende des zu transportierenden Produktes warten muss. Schwarze Trennstriche dienen der deutlicheren Darstellung zweier aufeinander folgender Operationen eines Auftrags auf derselben Maschine. Des Weiteren dienen sie der deutlicheren Darstellung zweier aufeinanderfolgender, beladener Fahrten eines FTS. Dieser Fall tritt ein, wenn ein FTS ein Produkt an einer Maschine ablädt und zugleich ein anderes Produkt an dieser für den Weitertransport aufnimmt. Die vierte und fünfte Transportfahrt von *AGV2* sind ein Beispiel hierfür. Der Endzeitpunkt einer Transportfahrt ist des Weiteren nicht zwangsläufig identisch mit dem Beginn der Bearbeitung des transportierten Produktes. Begründet liegt dies darin, dass das Produkt an einem Pufferplatz der Maschine abgelegt werden kann. In dem vorliegenden Fall werden die Produkte nach der Durchführung aller Operationen in ein Lager transportiert. Deshalb sind Transportfahrten auch nach der Fertigstellung der letzten Operation im gezeigten Ablaufplan vorhanden.

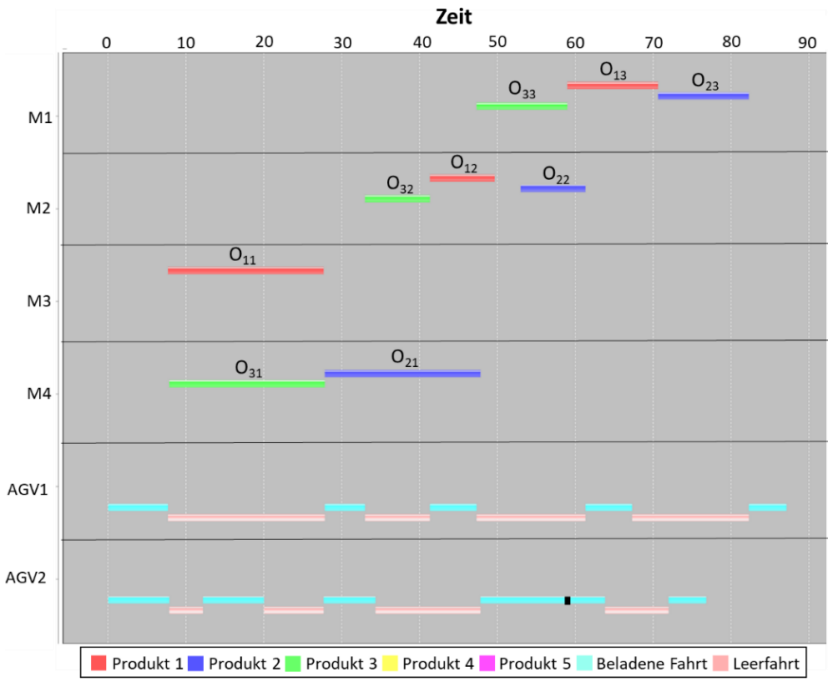


Abbildung 22: Gantt-Diagramm zur Repräsentation eines gemeinsamen Ablaufplans für Maschinen und FTS

2.4.3 Stand der Forschung zur simultanen Ablaufplanung von Maschinen und FTS

Die erste Forschungsarbeit zur SAMF wurde von Bilge und Ulusoy [158] veröffentlichtet. In dieser werden Benchmarkinstanzen (BI) für die SAMF präsentiert. Diese bestehen aus vier unterschiedlichen

Layoutvarianten einer beispielhaften Fertigung mit insgesamt vier, jeweils unterschiedlich angeordneten Maschinen sowie einer L/U (Load/Unload) Station. Durch die verschiedenen Layouts ergeben sich unterschiedliche Fahrwege und somit Fahrzeiten zwischen den einzelnen Maschinen. Zudem werden zehn Auftragssätze vorgestellt. Diese bestehen aus jeweils mehreren Aufträgen, innerhalb derer bestimmte Operationen auf den vorhandenen vier Maschinen zu bearbeiten sind. Die DLZ dient als Optimierungskriterium innerhalb des Benchmarks. Eine genauere Beschreibung dieser BI erfolgt in Kapitel 4.2. Diese BI werden auch heute noch zum Vergleich verschiedener Lösungsansätze genutzt. Zur Lösung des Problems wird in [158] zusätzlich ein nicht lineares Mixed Integer Programming (MIP) Modell vorgestellt. Ein Überblick bzgl. weiterer Veröffentlichungen zur simultanen Ablaufplanung wird im Folgenden gegeben. Die Aufführung der Veröffentlichungen, welche für die Ablaufplanung bzgl. der Maschinen eine JSSP-Umgebung zugrunde legen erfolgt zuerst, bevor anschließend die Aufführung solcher die eine FJSSP-Umgebung betrachten erfolgt.

Nageswararao et al. [159] schlagen einen Binary Particle Swarm Vehicle Heuristic Algorithmus (BPSVHA) für die SAMF vor. Der Algorithmus wird innerhalb der Studie mit anderen metaheuristischen Algorithmen verglichen. Erol et al. [160] schlagen ein in JADE implementiertes MAS für die dynamische und simultane Ablaufplanung vor. Das vorgeschlagene System wird gegen fünf Optimierungsalgorithmen für deterministische Fälle getestet. Gegenüber den zum Vergleich herangezogenen Optimierungsalgorithmen werden fast ausschließlich schlechter Ergebnisse bzgl. der DLZ erreicht. Die Untersuchung der Reaktion des vorgestellten Ansatzes auf dynamische Ereignisse ist nicht Bestandteil der Veröffentlichung. Der Ansatz wird zudem verschiedenen Konstruktionsverfahren gegenübergestellt, wobei das vorgeschlagene MAS in den meisten Fällen eine bessere DLZ erreicht. Mousavi et al. [161] präsentieren einen hybriden Algorithmus, bestehend aus GA und Partikelschwarmoptimierung zur SAMF. Die Minimierung der DLZ und der Anzahl an benötigten FTS werden als multikriterielle Zielfunktion verwendet. Eventuell notwendige Batterieladevorgänge der FTS werden während der Optimierung ebenfalls berücksichtigt. Chaudhry et al. [162] stellen einen GA zur Lösung der SAMF vor und testen diesen an den BI von Bilge und Ulusoy. Fontes und Homayouni [163] nutzen ein Mixed Integer Linear Programming (MILP) Modell, welches in der kommerziellen Software *Gurobi* implementiert wurde. Nachteil dieser Methode ist die teils sehr hohe RZ zur Findung einer Lösung. Lacomme et al. [164] präsentieren einen modifizierten, disjunktiven Graphen zur Modellierung des simultanen Ablaufplanungsproblems, sowie einen memetischen Algorithmus zur Lösung der SAMF. Die Verifizierung des Ansatzes erfolgt ebenfalls an den BI von Bilge und Ulusoy. Fauadi und Murata [165] schlagen eine Binary Particle Swarm Optimization (BPSO) vor, um das simultane Ablaufplanungsproblem zu lösen. Der Ansatz wird an den BI von Bilge und Ulusoy getestet und mit den Ergebnissen anderer Studien verglichen, wobei die Durchschnittswerte aller Instanzen pro Layout zum Vergleich herangezogen werden. Die vorgestellte BPSO erzielte in allen vier Layoutvarianten ein besseres Ergebnis, als die Ergebnisse des von Bilge und Ulusoy vorgestellten Ansatzes [158].

Im Gegensatz zu den, bis hierher vorgestellten Veröffentlichungen, legen die folgenden Veröffentlichungen zur SAMF eine FJSSP - anstatt einer JSSP-Umgebung zugrunde. Die erste Studie zur SAMF im Kontext einer FJSSP-Umgebung wurde von Deroussi und Norre [166] veröffentlicht. In dieser werden neue BI für das Ablaufplanungsproblem vorgestellt, welche auf bekannte FJSSP BI zurückgreifen, und diese um verschiedene Layoutvarianten bzgl. der jeweils vorhandenen Maschinen erweitern. Eine Iterative Lokale Suche (ILS) wird als Lösungsansatz für die simultane Ablaufplanung vorgestellt. In der Veröffentlichung werden allerdings nur erste Ideen und Ansätze, aber keine Simulationsresultate veröffentlicht. Kumar et al. [167] stellen in ihrer Veröffentlichung eine Erweiterung der

Instanzen von Bilge und Ulusoy [158] um jeweils drei alternative Maschinen zur Durchführung einer Operation vor. Das Layout der Maschinenanordnung sowie die Anzahl der Maschinen bleiben identisch. Kumar et al. stellen für die Optimierung der SAMF eine Kombination aus Differential Evolution (DE) Algorithmus, zur Ablaufplanung der Maschinen, und einer *Vehicle assignment heuristic* zur anschließenden Zuordnung der FTS zu den sich ergebenden Fahraufträgen vor. Eine *Machine selection heuristic* überprüft im abschließenden Schritt, ob die Nutzung einer alternativen Maschine für eine der Operationen zu einer Verbesserung der DLZ führt. Diese drei Prozessschritte werden iterativ wiederholt. Sahin et al. [168] stellen ein MAS für die SAMF vor, die auf dem Ansatz von Erol et al. [160] basiert und diesen von einer JSSP- auf eine FJSSP-Umgebung erweitert. Das System ist mit Prometheus entwickelt und in der JACK-Umgebung [169] programmiert. Die Erstellung des Ablaufplans erfolgt durch Verhandlung zwischen den einzelnen Agenten unter Nutzung des Kontraktnetzansatzes. Es handelt sich um einen rein dezentralen Ansatz ohne zentrale Steuerungseinheit. Im Vergleich zu den Ergebnissen aus [167] liefert das MAS von Sahin et al. in den meisten BI eine wesentlich schlechtere DLZ und in nur wenigen Fällen gleiche oder bessere DLZ. Das MAS ist in der Lage auf unerwartet auftretende Ereignisse zu reagieren. Diese Reaktionsfähigkeit geht entsprechend mit einer schlechteren Optimierungsqualität einher, wofür u. a. das Fehlen einer zentralen Einheit verantwortlich sein kann. Neu hinzugekommene Aufträge werden als Beispiel für unerwartet auftretende Ereignisse untersucht. Die neuen Aufträge werden erst nach vollständiger Bearbeitung der aktuellen Aufträge eingeplant und nicht in den aktuellen Ablaufplan integriert. Lin et al. [170] stellen in ihrer Arbeit einen simulationsbasierten Optimierungsansatz für die SAMF mit variierenden Bearbeitungszeiten vor. Sie verwenden eine Kombination aus lokaler Suche und GA, um gute Designalternativen bzgl. des Ablaufplans zu erforschen. Diese Designalternativen werden durch eine Simulation erzeugt. Neben variierenden Bearbeitungszeiten wurden auch andere unerwartet auftretende Ereignisse wie Stau und Stillstand der FTS berücksichtigt. Lin et al. verwenden verschiedene BI zur Validierung ihres Ansatzes. Zhang et al. [87], [171] lösen die simultane Ablaufplanung durch die Nutzung von Dekompensationsmethoden. In [87] wird ein GA verwendet, um jeder Operation eine Maschine und jedem Transportauftrag ein FTS zuzuordnen. Anschließend findet und verbessert eine Tabu-Suche (TS) eine Abfolge von Bearbeitungs- und Transportvorgängen für jede Maschine bzw. jedes FTS. Dieser Ansatz wird in [171] durch die Einbeziehung eines Shifting Bottleneck (SBN)-Verfahrens erweitert. Der GA wird, wie bisher, für die Zuordnung bzgl. der Operationen und Transportaufträgen zu den vorhandenen Maschinen und FTS verwendet. Das SBN-Verfahren findet bei jeder Generation des GA einen Ablaufplan, welchen die TS anschließend versucht, zu optimieren. Simulationsstudien mit den von Deroussi & Norre vorgeschlagenen BI werden präsentiert [166]. Für sechs Instanzen finden beide Methoden die gleiche DLZ; während jeder Ansatz in zwei der vier verbleibenden Instanzen besser als der andere ist. Deroussi [172] schlägt eine Hybridisierung einer Partikelschwarmoptimierung (PSO) mit einer lokalen Suche vor. Die PSO wird verwendet, um das Zuordnungs- und Ablaufplanungsproblem bzgl. der FTS zu lösen, sowie anschließend jeder Operation die zeitlich gesehen, erste verfügbare Maschine zuzuordnen. Basierend darauf erfolgt die Sequenzierung der Operationen auf den Maschinen, in dem diese, entsprechend ihrer Ankunftszeit im Maschinenpuffer, geordnet werden. Abschließend wird versucht, die erhaltene Lösung, durch lokales Suchverfahren weiter zu verbessern. Das Verfahren erreicht in der durchgeführten Simulationsstudie allerdings schlechtere Ergebnisse, als die Verfahren aus [87], [171]. Nouri et al. [173] stellen eine hybride Metaheuristik vor, welche aus einer Kombination von GA und TS besteht. Der GA dient zur Erstellung von Lösungen und die TS wird zur anschließenden Verbesserung dieser Lösungen verwendet. Die Autoren berichten über Verbesserungen der Ergebnisse bzgl. [87], [171] in sieben BI. Allerdings wird in [174] belegt, dass die Ergebnisse in [173] nicht plausible sind, und die

gefundenen Lösungen Reihenfolge- bzw. Ressourcenkapazitätsbedingungen verletzen. Begründet wird dies dadurch, dass die geringste mögliche DLZ des kürzesten Auftrages innerhalb der betroffenen Auftragssätze, bereits höher ist, als die in [174] vorgestellte DLZ für den gesamte Auftragssatz. Homayouni und Ponto [174] stellen ein MILP Modell zur Lösung der SAMF im Kontext eines FJSSP vor, und können in allen untersuchten BI gleichgute oder bessere Ergebnisse als [87] erreichen. Allerdings ist die dafür notwendige RZ gegenüber den zum Vergleich herangezogenen Optimierungsverfahren wesentlich höher. Dies schränkt den Einsatz dieses Verfahrens zur Steuerung einer realen Fertigung stark ein. In dieser Studie findet ebenfalls keine Berücksichtigung von Dynamik statt. Während der Erstellung dieser Arbeit (2020) wurden zwei auf CP-basierende Ansätze zur SAMF von Ham [175] (JSSP), [176] (FJSSP) vorgestellt. Keiner der beiden Ansätze ist allerdings in der Lage, auf unerwartet eintretende Ereignisse zu reagieren. Es wird lediglich eine deterministische Umgebung betrachtet.

Ein Überblick sowie eine Klassifizierung der vorgestellten Veröffentlichungen auf dem Gebiet der SAMF ist im Anhang K zu finden. Jede der vorgestellten Veröffentlichungen wird darin danach kategorisiert, ob unerwartet auftretende Ereignisse betrachtet werden, und ob darin eine JSSP- oder FJSSP-Umgebung vorliegt. In der Übersicht sind zudem noch einige weitere Veröffentlichungen zur SAMF in einer JSSP-Umgebung aufgeführt. Bzgl. der SAMF innerhalb einer FJSSP-Umgebung gibt es nach bestem Wissen des Autors und zum aktuellen Zeitpunkt, allerdings nur die zehn bereits präsentierten Veröffentlichungen. Die Anzahl der Werke, welche zusätzlich unerwartet auftretende Ereignisse und eine entsprechende Neuplanung betrachten, beschränkt sich auf die beiden Veröffentlichungen von Sahin et al. [168] sowie Lin et al. [170]. Keiner dieser beiden Ansätze ist in der Lage, Maschinenausfälle oder neue notwendige Operationen während der Prozessausführung zu berücksichtigen.

2.5 Forschungslücke und forschungsleitende Hypothese

Die Differenz zwischen der gegebenen Zielsetzung dieser Arbeit, sowie dem dargestellten Stand der Forschung, zeigt die zu schließende Forschungslücke auf. Diese lässt sich wie folgt untergliedern:

- In aktuellen Forschungsvorhaben werden zentrale oder dezentrale Ansätze der Produktionssteuerung untersucht. Diese sind jeweils mit Nachteilen behaftet, welche durch eine Kombination beider Ansätze minimiert werden könnten. Solche hybriden Ansätze zur Produktionssteuerung werden kaum untersucht [2].
- Die Berücksichtigung der Besonderheiten der Refabrikation in der Produktionssteuerung ist notwendig, da hier viele Herausforderungen auftreten, welche aus der Fertigung nicht bekannt sind [55], [57]–[61], [177]. Ein Ansatz zur Produktionssteuerung in der Refabrikation, inklusive der Berücksichtigung eines flexiblen Materialtransports, konnte weder als kommerzielle Software noch im Forschungsbereich gefunden werden.
- Eine SAMF findet in kommerziellen Systemen nicht statt und ist auch in der Literatur nur wenig untersucht. Veröffentlichte Untersuchungen beschäftigen sich zudem fast ausschließlich mit deterministischen, aber nicht mit praxisrelevanteren nicht deterministischen Umgebungen.
- Veröffentlichungen zur SAMF untersuchen, trotz der hohen Praxisrelevanz, nur unzureichend die Reaktionsfähigkeit auf unerwartet auftretende Ereignisse.
- Veröffentlichungen zur SAMF erreichen meist entweder gute Optimierungsergebnisse oder eine gute RZ. Allerdings sind für den Steuerungseinsatz in RS beide Kriterien zu erfüllen.
- Veröffentlichungen zur SAMF werden fast ausschließlich simulativ, nicht aber an der Implementierung in Use Case Szenarien validiert.

Die forschungsleitende Hypothese sowie die zur Validierung dieser zu beantwortenden Forschungsfragen sind in Abbildung 23 dargestellt.

Die Verwendung einer dezentral-hierarchischen Steuerungsarchitektur ermöglicht die Beherrschung variantenreicher und kundenindividueller Fertigung sowie die Berücksichtigung der Besonderheiten der Refabrikation. Die Kombination zentraler und dezentraler Elemente ermöglicht sowohl eine globale Optimierung, als auch die Reaktion auf unerwartet auftretende Ereignisse. Die Verwendung SAMF ermöglicht eine höhere Optimierungsqualität gegenüber einer sequentiellen Ablaufplanung.

Nummer	Forschungsfrage
1.	Bewirkt die SAMF eine DLZ-Reduktion gegenüber der sequenziellen sowie der selbstorganisierenden Ablaufplanung von Maschinen und FTS?
2.	Kann die SAMF, trotz der hohen Komplexität des Problems, in einer akzeptablen Rechenzeit erfolgen?
3.	Kann eine hybride Steuerungsarchitektur adäquat auf unerwartet auftretende Ereignisse reagieren, und gleichzeitig das Gesamtsystem global optimieren?
4.	Ist die hybride Steuerungsarchitektur für den industriellen Einsatz prinzipiell geeignet ?

Abbildung 23: Forschungsleitende Hypothese und Forschungsfragen

Die wissenschaftliche Vorgehensweise beinhaltet die Verifizierung des propagierten Ansatzes im Rahmen von Simulationsstudien. Dabei wird sowohl eine deterministische, als auch nicht deterministische Umgebung angenommen und untersucht. Darüber hinaus erfolgt ein Vergleich zu anderen veröffentlichten Ansätzen unter Verwendung standardisierter BI, um den Optimierungsalgorithmus gegenüber dem Stand der Forschung zu positionieren. Abschließend erfolgt die Validierung, also die Prüfung der praktischen Anwendbarkeit des vorgeschlagenen Lösungsansatzes, im Rahmen einer realen Fallstudie. Die Steuerungsarchitektur wird hierzu in einer Modellfabrik, zur Steuerung dieser, implementiert.

3 Konzept einer hybriden Architektur zur agilen Fertigungssteuerung

Die Anforderungen sowie das grundlegende Konzept der hybriden Steuerungsarchitektur werden in diesem Kapitel übersichtsartig erläutert. Die genaue Umsetzung wird in Kapitel 4 dargestellt. Eine Beschreibung der Implementierung des Konzeptes zur Steuerung einer Modellfabrik wird in Kapitel 5 gegeben.

3.1 Anforderungen

An die hybride Steuerungsarchitektur werden folgende Anforderungen gestellt:

- Eignung für (Re)fabrikationssysteme mit
  - hoher Variantenvielfalt bis hin zu Losgröße 1,
  - flexiblem Materialtransport und
  - einer Taktzeit von einigen Sekunden bis Minuten;
- Berücksichtigung des aktuellen Status sowie der Verfügbarkeit aller relevanten Fertigungsressourcen bei Steuerungsentscheidungen durch Echtzeit-Feedback von der Fertigungsebene;
- Integration und Berücksichtigung von Prozessplanänderungen eines Produktes in den laufenden Betrieb;
- Integration von FTS zur Realisierung eines flexiblen Materialtransports;
- Schnelle und adäquate Reaktion auf unerwartet auftretende Ereignisse;
- Simultane Optimierung der Ablaufplanung von Maschinen und FTS;
- Optimierungsdauer im Bereich von einigen Sekunden;
- Durchführung einer globalen Optimierung des RS.

Alle Anforderungen werden bei der Entwicklung der hybriden Steuerungsarchitektur berücksichtigt. Diese wird zuerst übersichtsartig erläutert und die einzelnen Komponenten im späteren Verlauf detailliert beschrieben.

3.2 Konzept der hybriden, dezentral-hierarchischen Steuerungsarchitektur

Das Konzept der hybriden Steuerungsarchitektur ist in Abbildung 24 dargestellt. Diese beinhaltet sowohl eine zentrale, als auch eine dezentrale Ebene und stellt nach der Klassifikation in Abbildung 3 eine dezentral-hierarchische Architektur dar. Sie besteht aus verschiedenen softwaretechnischen Komponenten und Komponententypen. Komponententypen sind Komponenten, von denen mehrere Instanzen erzeugt werden. Für jedes zu (re-)fabrizierende Produkt wird bspw. eine zugehörige Instanz einer Produkt Komponente erzeugt. Die Ressourcen -, Transport -, Experten - und Produkt Komponenten sind Komponententypen. Die Koordinator und Scheduler Komponente liegen hingegen einmalig vor. Die grundlegenden Funktionen aller Komponenten(-typen) sind folgend aufgelistet:

- *Koordinator Komponente*: Koordiniert die Ausführungsprozesse, die Ablaufplanung sowie die Interaktionen zwischen anderen Komponenten;
- *Scheduler Komponente*: Erstellt und optimiert die SAMF;

- *Ressourcen Komponenten:* Verwalten und steuern die Fertigungsressourcen (Maschinen und Handarbeitsplätze);
- *Produkt Komponenten:* Beinhaltet Fertigungsinformationen über das Produkt (Operationen, ausführbare Maschinen, Prozesszeiten, Reihenfolgebedingungen, …) und verwaltet den entsprechenden Auftrag innerhalb des Fertigungssystems;
- *Transport Komponenten:* Verwaltet und steuern die Transporteinheiten (FTS);
- *Experten Komponenten*: Berücksichtigung und Integration von Inspektionsergebnissen der Produkte innerhalb der Steuerung.

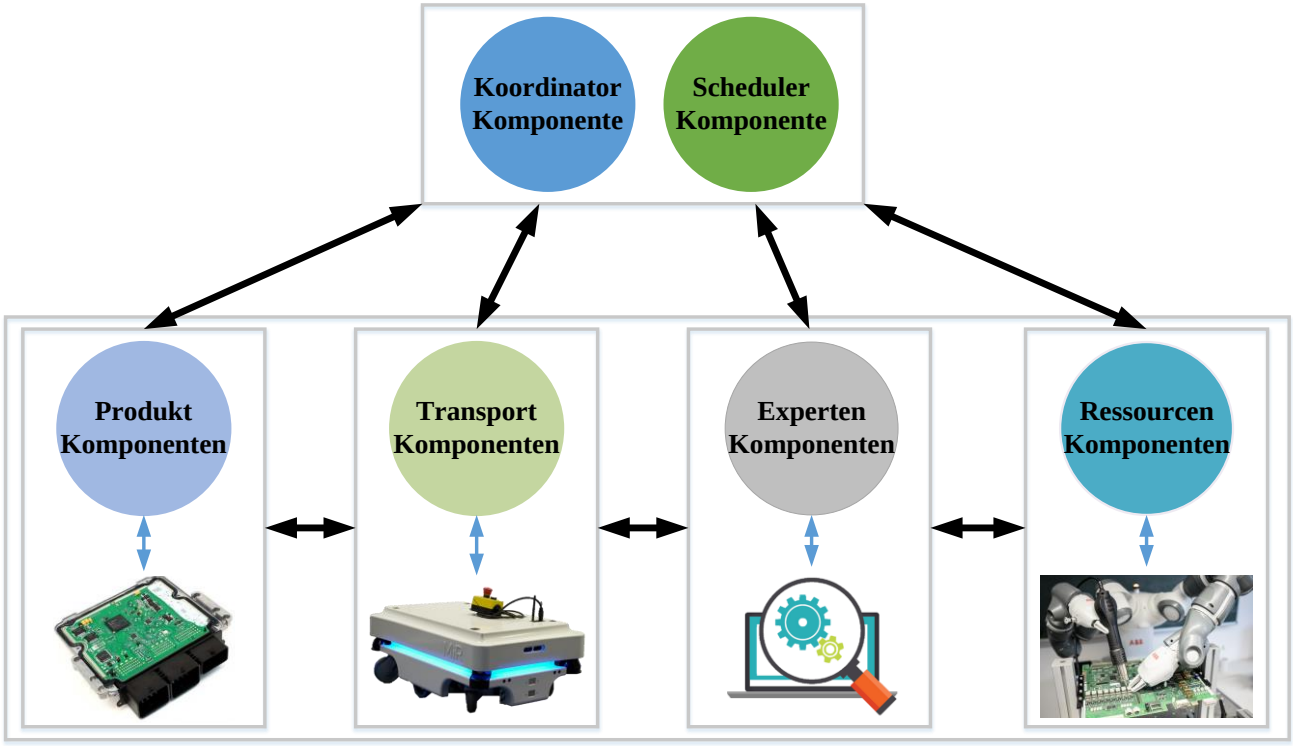


Abbildung 24: Hybride, dezentral-hierarchische Steuerungsarchitektur.

Alle Komponenten sind über geeignete Kommunikationskanäle in der Lage, miteinander zu kommunizieren. Ein Austausch von Informationen oder Steuerungsentscheidungen unter den einzelnen Komponenten ist somit möglich. Diese Fähigkeit erzeugt zusammen mit der Kopplung zwischen einigen Komponenten und physischen Fertigungsressourcen ein cyber-physisches System.

Die zentrale und die dezentrale Ebene besitzen jeweils unterschiedliche Aufgaben und Zuständigkeiten. Diese sowie ihre Zusammenhänge sind in Abbildung 25 dargestellt. Die Hauptaufgabe der zentralen Ebene besteht in der Erstellung und Optimierung der SAMF. Informationen über das Layout der Fabrik, die vorhandenen FTS und Fertigungsressourcen sowie über die Fahrzeiten der FTS zwischen den einzelnen Fertigungsressourcen werden hierzu benötigt. Die zu (re-)fabrizierenden Produkte dienen dem System als Eingangsgröße. Diese werden durch folgende Informationen parametrisiert:

- Zur (Re-)fabrikation des Produktes notwendige Operationen
- Reihenfolgebedingungen zwischen den Operationen
- Alle zur Bearbeitung einer Operation verfügbaren Maschinen
- Prozesszeit der Operationen in Abhängigkeit der bearbeitenden Maschine

Zur Durchführung der SAMF sind neben diesen statischen Daten auch Echtzeit-Daten von der Fertigungsebene notwendig. Dadurch kann der aktuelle Zustand des gesamten RS bei der SAMF berücksichtigt werden. Die Daten werden über Sensoren direkt von der Fertigungsebene aggregiert, durch die entsprechenden Komponenten interpretiert und anschließend an die zentrale Ebene weitergeleitet. Konkret handelt es sich bei diesen Daten bspw. um den Status sowie die Verfügbarkeit der einzelnen Fertigungsressourcen und FTS sowie den aktuellen Bearbeitungsstatus und die Position der sich im Umlauf befindlichen Produkte. Die Steuerungsarchitektur kann durch die kontinuierliche Überwachung dieser Daten unerwartet auftretende Ereignis detektieren. Sie kann in diesem Fall, unter Berücksichtigung des aktuellen Zustands, einen neuen gültigen Ablaufplan erstellen und entsprechende Steuerungseingriffe vornehmen. Die zentrale Ebene gibt den Ablaufplan nach Fertigstellung an die dezentrale Ebene weiter. Jede Komponente erhält den für sie relevanten Teil des Ablaufplans. Die Transport Komponenten der FTS initiieren im Anschluss den Transport der einzelnen Produkte zu den, für die Durchführung der jeweiligen Operationen ausgewählten, Stationen. Ein Produkt das von einem FTS an eine Station transportiert wurde, wird dort durch bspw. RFID-Technologie eindeutig identifiziert. Die Produkt Komponente ändert bei Ankunft an einer Station zudem den Status und den aktuellen Standort des Produktes. Zwei Typen von Stationen werden prinzipiell unterschieden:

- Bearbeitungsstationen: Bearbeitung einer Operation zur (Re-)fabrikation des Produktes
- Inspektionsstationen: Inspektion und anschließende Prozessplananpassung des Produktes

Die Ressourcen bzw. Experten Komponente der Station entscheidet dann, in Abhängigkeit des vorliegenden Produktes, über die passende Bearbeitungsroutine. Experten Komponenten repräsentieren und steuern Inspektionsstationen. Inspektionsstationen können die Inspektion manuell oder automatisiert durchführen. Ein Werker kann bspw. eine manuelle Sichtkontrolle durchführen und durch sein Expertenwissen entscheiden, welche weiteren Prozessschritte das Produkt zur Refabrikation durchlaufen muss. Eine Bildverarbeitungsapplikation kann die Inspektion alternativ automatisiert durchführen. Eine entsprechende Software analysiert die Ergebnisse und entscheidet, unter Zuhilfenahme von Informationen aus einer dezentralen Wissensbasis, über die weiteren, notwendigen Refabrikationsprozessschritte. Die dezentrale Wissensbasis ist jeweils Bestandteil der einzelnen Komponenten und umfasst das sie betreffende, notwendige Wissen. Mischformen dieser beiden Inspektionsverfahren sind ebenfalls möglich. Die Experten Komponente teilt der betroffenen Produkt Komponente, nach Abschluss der Inspektion, den neuen Prozessplan mit. Die Produkt Komponente stellt daraufhin eine Anfrage zur Neuplanung an die Koordinator Komponente. Anschließend fordert die Koordinator Komponente alle Produkt -, Transport -, Experten - und Ressourcen Komponenten auf, ihren aktuellen Status sowie ihre Verfügbarkeit der Scheduler Komponente mitzuteilen. Die Scheduler Komponente berücksichtigt diese Information bei der Durchführung der Neuplanung der SAMF. Die Integration von Inspektionsstationen ist innerhalb der Steuerungsarchitektur speziell für die Domäne Refabrikation notwendig. Die beschriebene Routine zur Neuplanung wird ebenfalls initiiert, wenn von einer der dezentralen Komponenten ein unvorhergesehenes Ereignis, wie bspw. der Ausfall einer Maschine, festgestellt wird.

Erreicht ein Produkt eine Bearbeitungsstation, bestimmt die entsprechende Ressourcen Komponente, unter Nutzung einer Wissensbasis, wie das vorliegende Produkt zu bearbeiten ist. Das Ergebnis hieraus ist eine Bearbeitungsprozedur. Eine Wissensbasis kann bspw. produktabhängige Bearbeitungsprogramme beinhalten, aber auch die automatische Erstellung von Bearbeitungsprogrammen umfassen,

wie es bspw. von Jungbluth [178] für die Roboter-assistierte Demontage vorgestellt wird. Die Ressourcen Komponente steuert im nächsten Schritt das zugehörige physische System, bspw. eine CNC-Maschine, oder einen Roboter, entsprechend dieser Prozedur. Sie informiert die Produkt Komponente des in Bearbeitung befindlichen Produktes über das Starten und Beenden der Bearbeitung. Diese kann dadurch den Status der aktuell durchgeführten Operation aktualisieren.

Ein Objekt, das sich auf dem geplanten Pfad befindet und diesen blockiert, löst eine Pfadneuplanung aus. Das betroffene FTS bzw. die zugehörige Transport Komponente führt die Pfadneuplanung dezentral durch. Die zentrale Ebene greift bei dieser Neuplanung nicht ein.

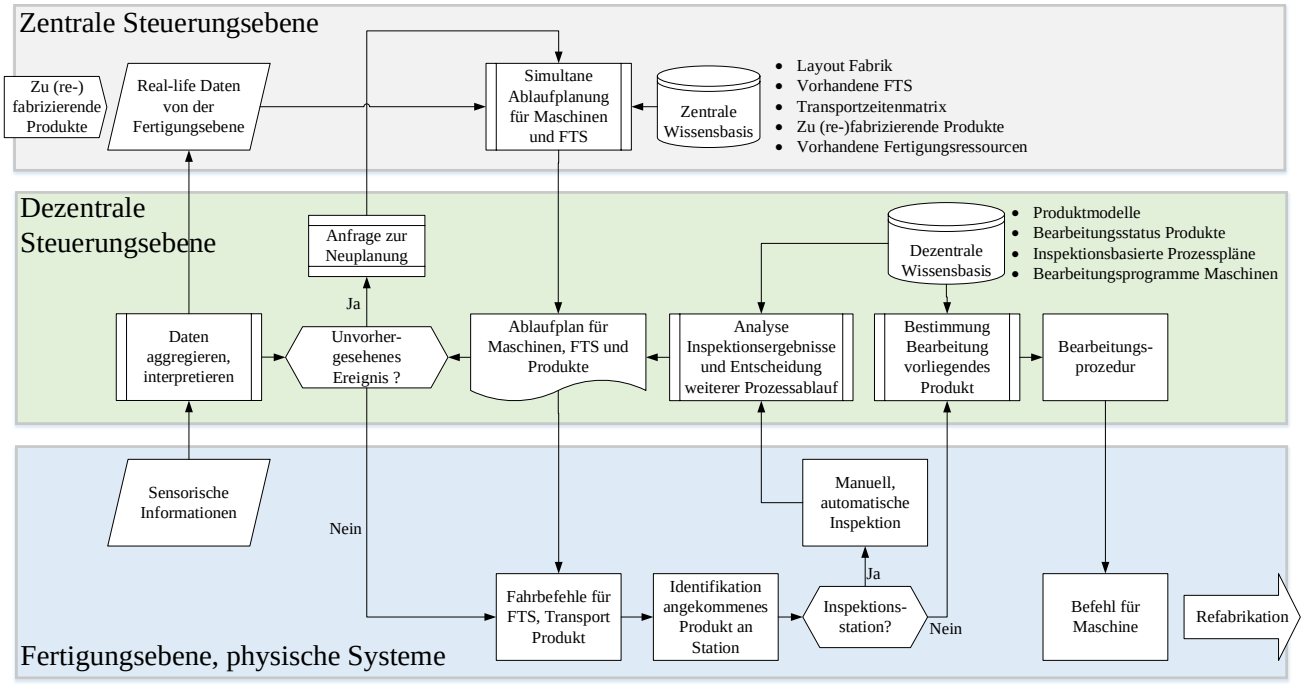


Abbildung 25: Aufgaben und Zuständigkeiten der einzelnen Ebenen der hybriden Steuerungsarchitektur.

Die dezentrale Ebene ist durch ihre beschriebene Autonomie in der Lage, auch bei kurzzeitigen Ausfällen der zentralen Ebene oder parallel zu einer Neuplanung mit der Ausführung der (Re-)fabrikation fortzufahren. Sie kann den vorliegenden Ablaufplan solange fortführen, bis dieser abgearbeitet ist, oder ein unerwartetes Ereignis eintritt, welches zur Behebung Aktionen der zentralen Ebene erfordert.

3.2.1 Vertikale und horizontale Integration der Steuerungsarchitektur

Zur Nutzung der hybriden Steuerungsarchitektur in einem RS ist eine vertikale und horizontale Integration dieser in die bestehende Leittechnik notwendig. Die vertikale Integration beschreibt die Eingliederung der Steuerungsarchitektur in die verschiedenen Hierarchieebenen der Automatisierungspyramide (Abbildung 4). Die horizontale Integration beinhaltet den Zusammenschluss verschiedener, sich auf derselben Automatisierungsebene befindlicher, Komponenten.

Die hybride Steuerungsarchitektur kann als Modul zur Feinplanung und Produktionssteuerung in die bestehende Leittechnik integriert werden. Die zentrale Ebene der Architektur übernimmt die Ablauf- und Ressourcenplanung und ist somit in die Betriebsleitebene einzugliedern. Alle dezentralen Komponenten sind in der Prozessleitebene einzugliedern. Die Ressourcen Komponenten übernehmen bspw. die Rezeptverwaltung sowie die Bedienung und Beobachtung der zugehörigen Ressourcen, was

eine Aufgabe dieser Automatisierungsebene ist. Das übergeordnete ERP-System sendet die Produktionsgrobplanung an die Koordinator Komponente. Diese benötigt dabei folgende Informationen:

- Anzahl der zu (re-)fabrizierenden Produkte
- Anzahl der vorhanden Bearbeitungs- und Inspektionsstationen
- Notwendige Operationen zur (Re-)fabrikation des jeweiligen Produktes
- Anzahl möglicher Bearbeitungs- oder Inspektionsstationen zur Durchführung einer Operation
- Mögliche Bearbeitungs- oder Inspektionsstationen zur Durchführung einer Operation
- Prozesszeit einer Operation (in Abhängigkeit der durchführenden Bearbeitungs- oder Inspektionsstation)
- Anzahl der vorhanden FTS
- Fahrzeiten zwischen den Bearbeitungs-, Lager- und Inspektionsstationen

Die Fahrzeiten zwischen den Bearbeitungs-, Lager- und Inspektionsstationen werden in Form einer Transportzeitmatrix dargestellt. Diese wird im .data-Format an die Steuerungsarchitektur übergeben (siehe Kapitel 4.1.1). Alle weiteren aufgeführten Informationen werden von dem übergeordneten ERR - oder einem anderweitigen System ebenfalls in Form einer .data-Datei benötigt. Ein Beispiel hierzu ist in Abbildung 26 dargestellt. Auf der linken Seite sind die Rohdaten zu sehen und rechts eine Erläuterung dieser. Die erste Zeile beinhaltet die Anzahl der Produkte des Auftragsatzes sowie der vorhandenen Maschinen (Summe aus Bearbeitungs- und Inspektionsstationen). Jede weitere Zeile repräsentiert den Auftrag eines zu (re-)fabrizierenden Produktes. In der ersten Spalte ist die Anzahl an notwendigen Operationen zu finden. Die nächste Spalte gibt an wie viele Maschinen die erste Operation durchführen können. In der folgenden Spalte sind alle Paare aus bearbeitender Maschine und zugehöriger Prozesszeit für diese Operation aufgelistet. Der erste Werte eines jeden Paars repräsentiert die Maschine (3 = Maschine 3) und der zweite Wert die zugehörige Prozesszeit in Sekunden. Diese Spaltenkombination aus Anzahl möglicher Maschinen zur Bearbeitung der Operation und anschließend den Paaren aus bearbeitender Maschine und zugehöriger Prozesszeit folgt für jede Operation eines Auftrags.

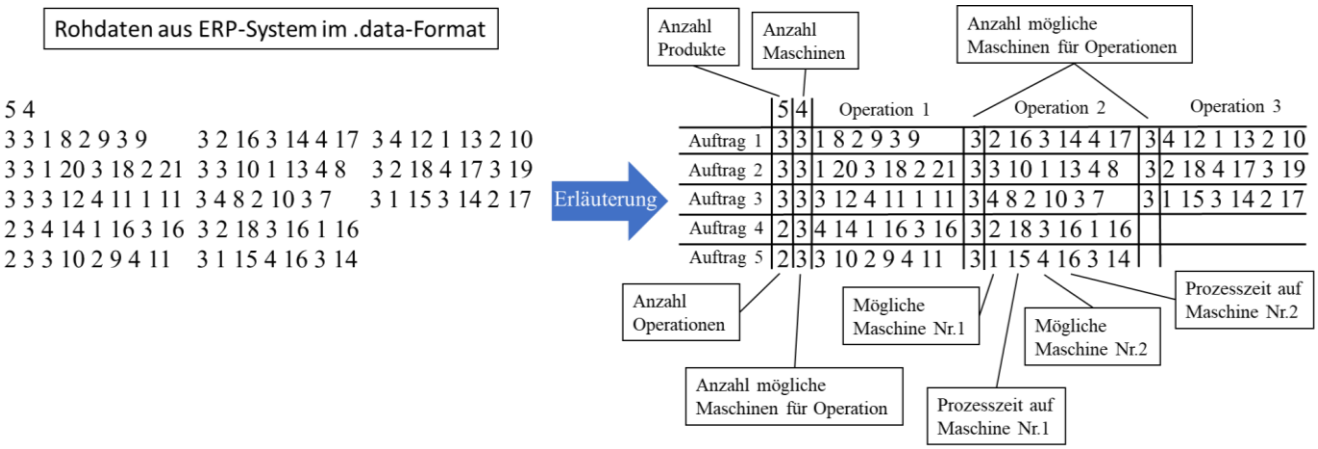


Abbildung 26: Format zur Datenübertragung von übergeordnetem System an die hybride Steuerungsarchitektur.

Die Steuerungsarchitektur übernimmt anschließend alle weiteren Automatisierungsaufgaben bis zur Steuerungsebene. An diese sendet die Steuerungsarchitektur entsprechende Befehle zur Ausführung bestimmter Routinen und Steuerungsaufgaben.

Die horizontale Integration findet hauptsächlich in der dezentralen Ebene statt. Hier müssen alle Produkt -, Transport -, Ressourcen - und Experten Komponenten miteinander kommunizieren können. Diese wird über eine geeignete, gemeinsame Kommunikationsschnittstelle ermöglicht.

Nach der groben Vorstellung des Konzepts der hybriden Steuerungsarchitektur werden folgend die einzelnen Komponenten und Komponententypen genauer beschrieben.

3.2.2 Koordinator Komponente

Die Koordinator Komponente hat die Aufgabe, die Kommunikation und Interaktion zwischen den einzelnen Komponenten zu koordinieren. Diese nutzt bei ihrer Entscheidungsfindung die von den anderen Komponenten gesammelten und von der Fertigungsebene aggregierten Daten. Sie ist zudem die Schnittstelle zwischen der hybriden Steuerungsarchitektur und dem übergeordneten PPS. Von diesem erhält die Koordinator Komponente die zu bearbeitenden Aufträge mit den entsprechenden Operationen, zur Bearbeitung möglichen Maschinen, Prozesszeiten sowie Reihenfolgerandbedingungen. Die Definition der Schnittstelle zwischen Steuerungsarchitektur und PPS wurde in Abbildung 26 bereits gegeben. Die Hauptmerkmale der Koordinator Komponente sind in Tabelle 1 zusammengefasst.

Zweck	Koordination der Kommunikation und Interaktionen zwischen den verschiedenen Komponenten;	
	Entscheidung über die Durchführung einer (Neu-)planung.	
Daten	Regeln, die das Verhalten des Systems bei Ereignissen wie Auftragseingang, Maschinenausfall, etc. steuern;	
	Parameter, die den Zeitpunkt der (Neu-)planung definieren.	
Aktionen	Reaktion in „Echtzeit“ auf Zustandsänderungen im RS;	
	Andere Komponenten über die Zustandsänderungen informieren;	
	Instanziieren von Komponenten für neue zu (re-)fabrizierende Produkte	
Ausgehend	Produkt Komponenten	Informationen über neue Aufträge;
	Scheduler Komponenten	Benachrichtigung zur Ausführung einer (Neu-)planung;
		Benachrichtigung mit Aufforderung zur Veröffentlichung des neuen Ablaufplans.
Eingehend	Ressourcen Komponenten	Informationen über Statusänderungen der Ressourcen;
		Informationen über die Verfügbarkeit der Ressourcen.
	Transport Komponenten	Informationen über Statusänderungen der FTS;
		Informationen über die Verfügbarkeit der FTS.
	Produkt Komponenten	Informationen über inaktive Bearbeitungsmöglichkeiten für die bestehenden Produkte (durch Maschinenstörung);
		Informationen über die Fertigstellung eines Produktes.
	Experten Komponenten	Information über die Modifikation des Ablaufplans eines Produktes durch zusätzlich notwendige Operation(en).
	Scheduler Komponente	Benachrichtigung über die Fertigstellung der Ablaufplanung.

Tabelle 1: Merkmale der Koordinator Komponente

Die Koordinator Komponente erhält durch die Nutzung und Analyse der Daten aller anderer Komponenten ein Gesamtbild über das RS. Sie kann dadurch unerwartet auftretende Ereignisse innerhalb des RS detektieren und die einzelnen Komponenten der Steuerungsarchitektur zur adäquaten Reaktion auf diese koordinieren. Bekommt die Koordinator Komponente den Ausfall einer Maschine oder die Notwendigkeit zum Laden der Batterie eines FTS von der betroffenen Ressourcen respektive Transport

Komponente übermittelt, stößt diese eine Neuplanung bei der Scheduler Komponente an. Die Koordinator Komponente kann bei der Ankunft eines neuen Auftrags zudem entscheiden, ob eine Neuplanung sofort, oder erst wenn eine gewisse Anzahl neuer Aufträge vorhanden ist, ausgelöst wird. Alternativ ist auch eine zyklische Neuplanung möglich. Der Auftrag wird in diesem Fall bei der nächsten zyklischen Neuplanung in der Ablaufplanung mitberücksichtigt.

3.2.3 Scheduler Komponente

Die Scheduler Komponente erstellt bzw. optimiert den Ablaufplan für das RS und übernimmt sowohl die initiale Ablaufplanung, als auch die Neuplanung beim Auftreten unerwarteter Ereignisse. Sie nutzt hierfür einen auf CP basierenden Algorithmus zur SAMF. Dieser wird in Kapitel 4.1 vorgestellt und erläutert. Als Zielfunktion erfolgt die Nutzung der DLZ, der durchschnittlichen MA, oder eine Kombination beider Kriterien in Form einer lexikografischen, multikriteriellen Optimierung. Die Scheduler Komponente berücksichtigt bei der Ablaufplanung den aktuellen Status sowie die Verfügbarkeit aller relevanten Fertigungsressourcen. Sie erhält diese Informationen sowie die benötigten Randbedingungen von den entsprechenden Ressourcen -, Experten -, Transport - und Produkt - Komponenten. Die Merkmale der Scheduler Komponenten sind in Tabelle 2 zusammengefasst.

Zweck	SAMF und Berücksichtigung des Status sowie der Verfügbarkeit aller relevanten Fertigungsteilnehmer;	
	Bereitstellung des aktuellen Ablaufplans an die anderen Transport -, Produkt -, Ressourcen - und Experten Komponenten.	
Daten	Aktueller Ablaufplan;	
	Parameter zur Definierung des Lösungsverfahrens auf Planungsebene;	
	Zielfunktion für die Ablaufplanung, z. B. DLZ und / oder durchschnittliche Maschinenauslastung.	
Aktionen	Erstellung und Optimierung eines initialen Ablaufplans zu (Re-)fabrikation der vorliegenden Produkte mittels der zur Verfügung stehenden Ressourcen und FTS;	
	Bei Aufforderung Neuplanung unter Berücksichtigung von Echtzeit-Daten von der Fertigungsebene;	
	Mitteilung der relevanten Teile des Ablaufplans an die Produkt -, Transport -, Experten - und Ressourcen Komponenten.	
Ausgehend	Ressourcen Komponenten	Relevante Teile des Ablaufplans.
	Produkt Komponenten	Relevante Teile des Ablaufplans.
	Transport Komponenten	Relevante Teile des Ablaufplans.
	Experten Komponenten	Relevante Teile des Ablaufplans.
	Koordinator Komponente	Mitteilung über die Fertigstellung der Ablaufplanung.
Eingehend	Ressourcen Komponenten	Status der Ressourcen und deren Verfügbarkeit.
	Experten Komponenten	Status der Ressourcen und deren Verfügbarkeit.
	Produkt Komponenten	Einzelne durchzuführende Operationen mit zugehörigen möglichen zur Bearbeitung geeigneten Maschinen und den jeweils entsprechenden Bearbeitungszeiten, Operationsreihenfolge, aktueller Stand der Auftragsbearbeitung, aktuelle Position des Produktes im RS.
	Transport Komponenten	Tatsächliche Fahrzeiten zwischen den einzelnen Maschinen;
		Status und Verfügbarkeit der FTS.
	Koordinator Komponente	Aufforderung zum Starten der (Neu-)planung;
		Liste der Aufträge, die in der Planung berücksichtigt werden müssen;

Tabelle 2: Merkmale der Scheduler Komponente

Die Scheduler Komponente erhält die Aufforderung zur Erstellung eines initialen Ablaufplans, bzw. zur Neuplanung von der Koordinator Komponente. Sobald die Scheduler Komponente die Aufforderung zur Durchführung einer (Neu-) Planung erhält, erfragt diese bei allen Transport -, Experten - und Ressourcen - Komponenten den jeweiligen Status sowie die Verfügbarkeit der entsprechenden Fertigungsteilnehmer ab. Von den Produkt Komponenten wird zudem Folgendes angefragt:

- Auszuführende Operationen
- Zur Bearbeitung einer jeden Operation verfügbare Maschinen
- Geltende Reihenfolgebedingungen zwischen den Operationen
- Aktuelle Position des Produktes innerhalb des RS
- Status der aktuell in Bearbeitung befindlichen Operation

Die Verfügbarkeiten von Ressourcen und FTS werden in der Neuplanung mit dem Zeitpunkt der Fertigstellung des jeweils aktuell laufenden Transportvorgangs, respektive der jeweiligen Bearbeitung, berücksichtigt.

Ist die (Neu-)planung erfolgreich abgeschlossen, wird die Koordinator Komponente darüber informiert. Den einzelnen Ressourcen -, Transport - und Produkt Komponenten werden die jeweils zugehörigen Teile des Ablaufplans übermittelt. Eine ausführlichere Beschreibung des Ablaufs einer Neuplanung innerhalb der Steuerungsarchitektur wird in Kapitel 4.1.3 gegeben.

3.2.4 Produkt Komponenten

Produkt Komponenten dienen der Modellierung und Repräsentation aller sich im Umlauf befindlichen Produkte sowie der dazugehörigen Aufträge. Sie beinhalten alle relevanten Informationen bzgl. des jeweiligen Auftrags sowie des aktuellen Bearbeitungsfortschritts. Jeder Produkt Komponente ist genau ein physisches zu (re-)fabrizierendes Produkt zugeordnet. Die Produkt Komponente passt den Status der einzelnen durchzuführenden Operationen bzgl. des Prozessfortschrittes kontinuierlich an. Sie verfolgt zudem die aktuelle Position des zugehörigen Produktes innerhalb des RS und gibt Informationen über den aktuellen Bearbeitungsstatus sowie die noch durchzuführenden Operationen auf Anfrage an andere Komponenten weiter. Die Hauptmerkmale einer Produkt Komponente sind in Tabelle 3 zusammengefasst dargestellt.

Zweck	Verwaltung und Nachverfolgung auftragsrelevanter Daten sowie des Bearbeitungsfortschritts.	
Daten	Informationen über den ausgewählten Prozessplan, dem die Operation des Auftrags folgen müssen;	
	Informationen über den aktuellen Bearbeitungsstatus des Produktes.	
Aktionen	Bereitstellung planungsrelevanter Informationen bzgl. des Produktes;	
	Sammeln und Bereitstellen von Informationen über den aktuellen Bearbeitungsstatus;	
	Schätzen der voraussichtlichen Fertigstellungszeit der aktuell in Bearbeitung befindlichen Operation.	
Ausgehend	Ressourcen Komponenten	Benachrichtigung der Ressourcen Komponenten über die Ankunft des Produktes;
	Scheduler Komponente	Informationen über die aktuelle Position des Produktes innerhalb des RS;
		Informationen über die noch durchzuführenden und damit in der Neuplanung zu berücksichtigenden, Operationen.
	Koordinator Komponente	Benachrichtigung über die Fertigstellung des Auftrags.

Eingehend	Ressourcen Komponenten	Informationen über Ankunft und Abfahrt an der Ressource sowie über Start und Ende der entsprechenden Bearbeitung.
	Experten Komponenten	Informationen über evtl. zusätzlich notwendige Operationen.
	Scheduler Komponente	Informationen über durchzuführende Operationen mit den möglichen Maschinen und den entsprechenden Bearbeitungszeiten sowie dem zugehörigen Ablaufplan.

Tabelle 3: Merkmale der Produkt Komponenten.

Die Koordinator Komponente instanziiert und parametrisiert für jedes neue Produkt das (re-)fabriziert werden soll eine neue zugehörige Produkt Komponente. Die Scheduler Komponente teilt der neuen Produkt Komponente den zugehörigen Ablaufplan mit, sobald das neue Produkt im Rahmen einer Neuplanung berücksichtigt wurde.

Das Produkt bzw. die Produkt Komponente meldet sich bei der Ankunft an einer Bearbeitungs- oder Inspektionsstation mittels automatischer Identifizierungs-Techniken (bspw. einem RFID-Chip) bei der zugehörigen Ressourcen respektive Experten Komponente an. Diese bestätigt der Produkt Komponente wiederum die Ankunft an der Maschine. Die Produkt Komponente ändert nach der Ankunft an einer Ressource den Status der aktuellen Operation von „in Transport“ zu „an bearbeitender Ressource“. Jede Operation durchläuft die folgenden fünf Zustände in der gegebenen Reihenfolge:

1. *„Unbearbeitet“*
2. *„In Transport“*
3. *„An bearbeitender Ressource“*
4. *„In Bearbeitung“*
5. *„Bearbeitung abgeschlossen“*

Das Zustandsübergangsdiagramm, welches die zur Auslösung der jeweiligen Zustandsänderung notwendigen Ereignisse aufzeigt, ist in Abbildung 27 dargestellt.

Die Produkt Komponente speichert den Zeitpunkt der Ankunft des Produktes an jeder Ressource und den Zeitpunkt des Beginns der Bearbeitung einer jeden Operation. Der Fertigstellungszeitpunkt der Bearbeitung sowie der Startzeitpunkt des nächsten Transports des Produktes werden ebenfalls gespeichert. Das Sammeln dieser Daten ermöglicht es, statistische Analysen bzgl. der tatsächlichen Bearbeitungs- und Liegezeiten durchzuführen. Die Analyse dieser Daten ist nicht Bestandteil der vorliegenden Arbeit. Die Historie der (Re-)fabrikation des Produktes kann durch das Speichern dieser Informationen ebenfalls nachvollzogen werden. Die Ressourcen Komponente bestätigt der Transport Komponente das erfolgreich Ab- bzw. Aufladen eines Produktes an einer Station. Erst wenn diese Bestätigung erfolgt ist kann das FTS mit seinem nächsten Transportauftrag beginnen. Dadurch wird sichergestellt, dass es erst nach erfolgreichem Ab- bzw. Aufladevorgang zu einer Weiterfahrt des FTS kommt.

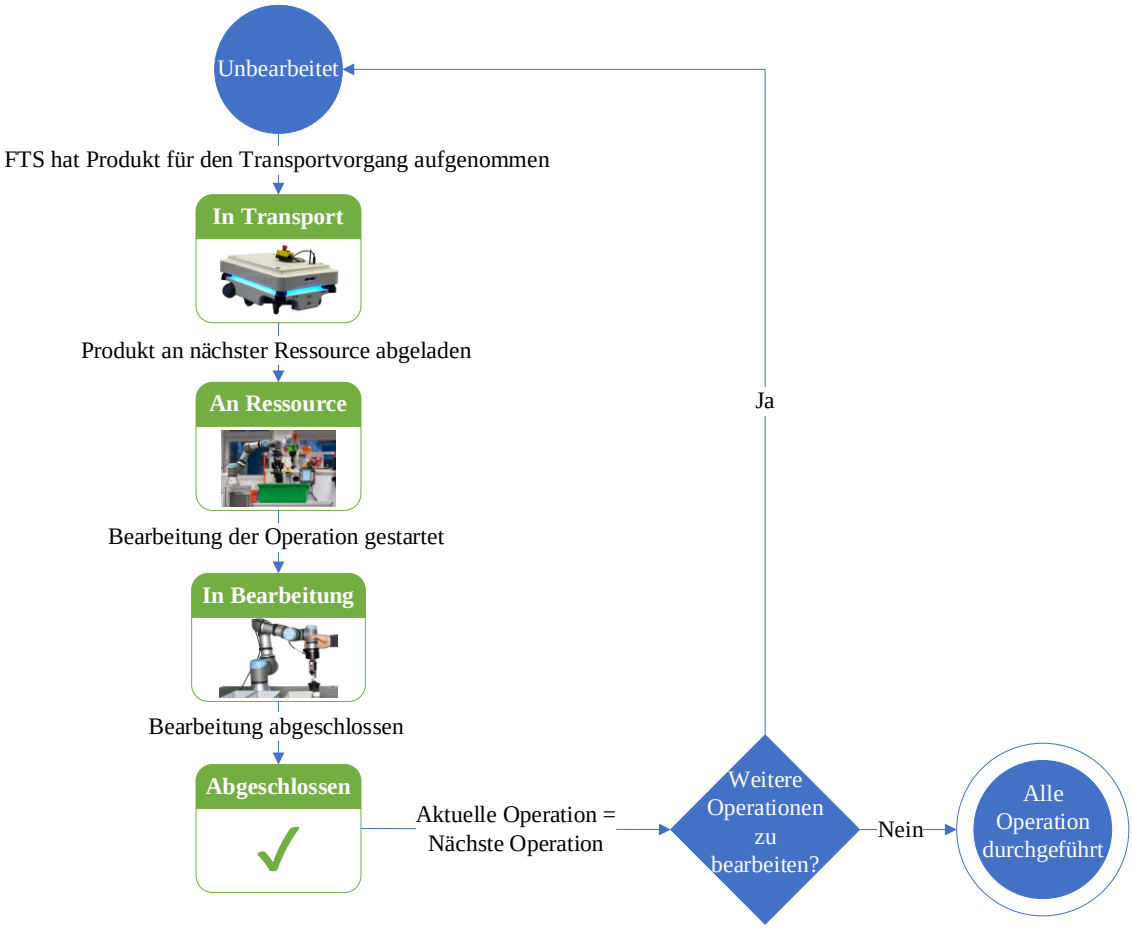


Abbildung 27: Zustandsübergangsdiagramm Produkt Komponente.

Eine Neuplanung kann zu jedem beliebigen Zeitpunkt stattfinden. Die Produkt Komponenten entscheiden in diesem Fall, welche Operationen sie zur Neuplanung an die Scheduler Komponente übermittelt und welche nicht. Sie können dadurch eine Neuplanung für bestimmte Operationen annehmen, oder auch ablehnen. Änderungen für Operationen, die sich gerade in Bearbeitung befinden, oder für Operationen, die sich aktuell in Transport zur zugeordneten Maschine befinden, lehnen die Produkt Komponenten ab. Dies gilt auch für Operationen die sich bereits an der sie bearbeitenden Maschine befinden. Diese Operationen werden entsprechend nicht an die Scheduler Komponente übermittelt, sondern die bisherige Planung für diese Operationen beibehalten. Allerdings wird die sich ergebende Verfügbarkeit, bis zum voraussichtlich frühestmöglichen Zeitpunkt zur Durchführung der Nachfolgeoperation sowie die aktuelle Position des Produktes, an die Scheduler Komponente übermittelt. Dieses Vorgehen reduziert bzw. vermeidet zusätzliche Transport- und Liegezeiten innerhalb des RS. Alle Operationen mit dem Status „Unbearbeitet“ werden an die Scheduler Komponenten zur Neuplanung übermittelt. Die Produkt Komponenten akzeptieren hiermit eine Neuplanung dieser Operationen. Sie erhalten anschließend den neuen Ablaufplan für diese Operationen von der Scheduler Komponente und ersetzten den alten Ablaufplan durch diesen. Abbildung 28 stellt diesen interner Prozess der Produkt Komponenten als Ablaufdiagramm dar.

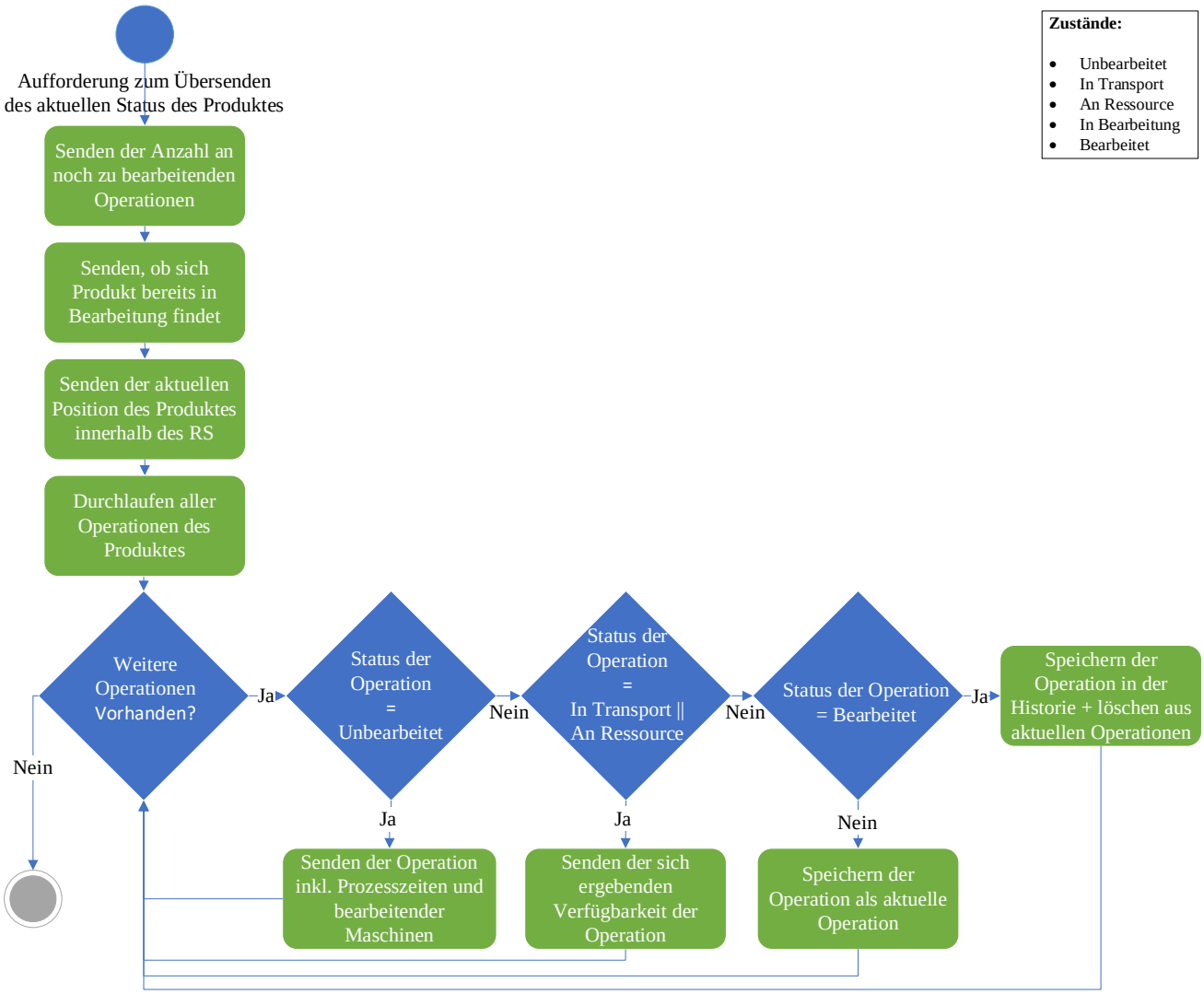


Abbildung 28: Ablaufdiagramm Produkt Komponente.

Nachdem alle Operationen eines Auftrags bearbeitet wurden und der Auftrag das RS verlassen hat, wird die zugehörige Produkt Komponente archiviert.

3.2.5 Ressourcen Komponenten

Jede Bearbeitungsstation wird durch eine Ressourcen Komponente digital abgebildet und gesteuert. Das digitale Abbild verfügt über Informationen zur aktuellen Auslastung, Pufferbestand, Status und Verfügbarkeit der Bearbeitungsstation sowie anderen relevanten Kennzahlen. Sind mehrere Fertigungsressourcen zur Bearbeitung einer einzigen Operation notwendig, werden diese als eine Ressourcen Komponente abgebildet. Beispielhaft hierfür kann ein Roboter sein, der eine CNC-Fräse bestückt und ein weiterer Roboter, der die CNC-Fräse nach Fertigstellung der Bearbeitung entlädt. Diese drei Ressourcen werden durch eine Ressourcen Komponente zusammengefasst, da die entsprechende Operation nur durch Zusammenwirken der drei Maschinen vollzogen werden kann. Die Hauptmerkmale der Ressourcen Komponenten sind in Tabelle 4 zusammengefasst.

Zweck	Steuerung von Fertigungsressourcen;
	Überwachung von Fertigungsressourcen.
Daten	Status der Ressource;
	Verfügbarkeit der Ressource;

	Aktuell in Bearbeitung befindliche Operation;	
	Belegung der an der Ressource vorhandenen Pufferplätze;	
	Aktueller Ablaufplan bzgl. der entsprechenden Ressource;	
Aktionen	Reaktion auf jede Zustandsänderung an der Ressource;	
	Informieren der anderen Komponenten über Zustandsänderungen;	
	Sammeln von Produktionsstatistiken und Weitergabe dieser an andere Komponenten.	
Ausgehend	Produkt Komponente	Informationen über alle Zustandsänderungen im Zusammenhang mit dem Produkt, z. B. Start und Ende der Bearbeitung;
	Koordinator Komponente	Benachrichtigung bei Störungen.
	Scheduler Komponente	Information über den Status der Ressource;
		Informationen über die Verfügbarkeit der Ressource;
		Information über die sich aktuell in Bearbeitung befindliche Operation.
Eingehend	Produkt Komponenten	Informationen über die Ankunft und Abholung des Produktes;
	Transport Komponenten	Informationen über die Ankunft und Abholung eines Produktes;
	Scheduler Komponente	Ablaufplan, der für diese Ressource relevant ist.
	Koordinator Komponenten	Aufforderung zur Übermittlung des aktuellen Status und der Verfügbarkeit.

Tabelle 4: Merkmale der Ressourcen Komponenten.

Ressourcen Komponenten vergleichen während der Bearbeitung einer Operation die tatsächliche Bearbeitungszeit, mit der ihr übermittelten, geschätzten Bearbeitungszeit. Eine Ressourcen Komponente schließt auf eine Störung, oder einen Ausfall, wenn die tatsächliche Prozesszeit um einen bestimmten Betrag höher als die geschätzte ist. Sie übermittelt in diesem Fall an die Koordinator Komponente, dass vermutlich eine Verzögerung, oder ein Ausfall vorliegt und sie aktuell nicht für eine Bearbeitung verfügbar ist.

Zudem wäre es möglich, dass die Ressourcen Komponenten alle tatsächlichen Prozesszeiten gleichartiger Operationen sammeln und analysieren, um eine genauere Aussage über die Prozesszeiten geben zu können. Die Arbeitsvorbereitung trifft alternativ eine Schätzung über die Prozesszeiten und teilt diese dem System mit. Eine Verwendung realitätsnaher Prozesszeiten erhöht die Qualität und Realisierbarkeit, der von der Scheduler Komponente erzeugten Ablaufpläne.

3.2.6 Transport Komponenten

Jedes FTS wird durch eine zugehörige Transport Komponente repräsentiert und gesteuert. Diese sammelt Informationen über das FTS. Der Status, die Verfügbarkeit, die aktuelle Position, der Ladezustand und das sich aktuell auf dem FTS befindliche Produkt sind Beispiele hierfür. Die Transport Komponente übernimmt die FTS-Steuerungsaufgaben Lokalisierung, Pfadplanung, Bewegungsplanung und Fahrzeugmanagement, die in Kapitel 2.4 erläutert wurden. Die Hauptmerkmale der Transport Komponenten sind in Tabelle 5 zusammengefasst.

Zweck	Steuerung der FTS;
	Überwachung der FTS.
Daten	2D-Karte der Umgebung;
	Aktueller Transportauftrag des FTS;
	Information über den Status des FTS;

	Information über die Verfügbarkeit des FTS;	
	Aktueller Ablaufplan bzgl. der Transportaufträge des FTS;	
	Liste an Transportaufträgen, die in der Warteschlange auf ihre Abarbeitung warten.	
Aktionen	Pfad- und Bewegungsplanung;	
	Pfad- und Bewegungsneuplanung beim Auftreten unerwarteter Objekte auf dem geplanten Pfad;	
	Informieren der anderen Komponenten über Statusänderungen (z. B. beginnender Batterieladevorgang).	
Ausgehend	Produkt Komponenten	Informationen über alle Zustandsänderungen im Zusammenhang mit den Transportaufträgen, z. B. Start und Ende der Transportfahrten.
	Koordinator Komponente	Benachrichtigung über alle Statusänderungen des FTS.
	Scheduler Komponente	Aktueller Status, Verfügbarkeit und Position des FTS;
		Information über den aktuell in Ausführung befindliche Transportauftrag.
Eingehend	Scheduler Komponente	Ablaufplan, der für das jeweilige FTS relevant ist.

Tabelle 5: Merkmale der Transport Komponenten.

Die Transport Komponenten übermitteln an die Scheduler Komponente, im Falle einer Neuplanung, alle Informationen, die zur Erstellung des neuen Ablaufplans notwendig sind. Diese Informationen umfassen:

- Die aktuelle Position des FTS
- Den Status des FTS:
  - Verfügbar (Leerlauf)
  - Aktuell Durchführung einer Leerfahrt
  - Aktuell Durchführung einer beladenen Fahrt
  - Aktuell Durchführung eines Ladevorgangs
- Voraussichtlicher Zeitpunkt, zu dem das FTS den beladenen Transport oder den Ladevorgang abgeschlossen hat und somit für den nächsten Transportauftrag wieder verfügbar ist.

Die Transport Komponente lässt, im Rahmen einer Neuplanung, nur die Unterbrechung von Leerfahrten zu, nicht aber von beladenen Fahrten. Die Unterbrechung einer beladenen Fahrt kann zu Komplikationen führen, da das geladene Produkt zuerst an einem Zwischenpuffer abgeladen werden muss, bevor das nächste Produkt aufgenommen werden kann. Die Transport Komponente setzt ihre Verfügbarkeit hierzu auf den Zeitpunkt fest, an dem die aktuelle beladene Fahrt voraussichtlich abgeschlossen sein wird. Sie übermittelt den jeweiligen Verfügbarkeitszeitpunkt an die Scheduler Komponente, welche diese Information im Rahmen der Neuplanung mitberücksichtigt. Die Scheduler Komponente übermittelt nach der Neuplanung den jeweils zugehörigen Ablaufplan an die Transport Komponenten. Diese ersetzen ihren bisher gültigen Ablaufplan durch den neu erhaltenen.

3.2.7 Experten Komponenten

Experten Komponenten repräsentieren und steuern Inspektionsstationen. Diese inspizieren die zu refabrizierenden Produkte bzgl. Verschmutzungen, Funktionalitäten, Toleranzen, oder anderweitige Beschädigungen. Sie treffen anschließend, basierend auf Expertenwissen, eine Entscheidung über den weiteren notwendigen Prozessablauf zur Refabrikation des vorliegenden Produktes. Dadurch können sich zusätzlich zu durchlaufende Operationen für das inspizierte Produkt ergeben. Die Durchführung

der Inspektion kann über eine automatisierte, physische Prüfstation erfolgen. Alternativ kann auch ein Werker die Entscheidung über den weiteren Prozessablauf treffen und diese der Experten Komponente durch eine Mensch-Maschine-Schnittstelle (MMS) mitteilen. Dieser inspiziert hierzu zuerst das Produkt und trifft anschließend, basierend auf seinem Expertenwissen, eine Entscheidung über den weiteren Prozessablauf. Die Experten Komponente kommuniziert die jeweilige Entscheidung über den weiteren Prozessablauf in beiden Fällen an die betroffenen Produkt Komponente. Zudem wird die Koordinator Komponente darüber informiert, dass Änderungen an einem Prozessplan vorgenommen wurden. Diese initiiert daraufhin eine Neuplanung, um die Prozessplanänderung ablaufplantechnisch zu berücksichtigen. Die Integration von Experten Komponenten ist speziell in der Domäne Refabrikation notwendig. Die Hauptmerkmale von Experten Komponenten sind in Tabelle 6 zusammengefasst.

Zweck	Steuerung von Inspektionsstationen;	
	Integration durch den Werker getroffener Änderungen am Prozessablauf eines Produktes;	
	Entscheidung über den weiteren Prozessablauf der Produkte, basierend auf den Inspektionsergebnissen.	
Daten	Inspektionsroutinen und –programme;	
	Wissensbasis zur Bestimmung des weiteren Prozessplans basierend auf den Inspektionsergebnissen.	
Aktionen	Inspektion der Produkte;	
	Analyse der Inspektionsergebnisse;	
	Entscheidung, wie das Produkt alternativ bearbeitet werden soll;	
	Benachrichtigung betroffener Produkt Komponenten über die Änderungen.	
Ausgehend	Koordinator Komponente	Benachrichtigung, dass Änderungen an dem Ablaufplan eines Produktes vorgenommen wurden;
		Aufforderung zur Neuplanung.
	Produkt Komponenten	Geänderter Ablaufplan mit den evtl. zusätzlich notwendigen Operationen.
Eingehend	Produkt Komponente	Information über das aktuell vorliegende Produkt;
	Transport Komponente	Benachrichtigung über Ankunft eines neuen Produktes.

Tabelle 6: Merkmale der Experten Komponenten.

Nachdem in diesem Kapitel das grundlegende Konzept der hybriden Steuerungsarchitektur sowie die Aufgaben und Funktionen der einzelnen Komponenten und Komponententypen innerhalb dieser präsentiert wurden, erfolgt in Kapitel 4 eine detaillierte Beschreibung der Steuerungsarchitektur.

4 Hybride Steuerungsarchitektur für (Re-)fabrikationssysteme

In diesem Kapitel werden die Steuerungsarchitektur detaillierter beschrieben und Simulationsstudien zur Verifizierung dieser vorgestellt. Der Fokus liegt hierbei auf der Scheduler Komponente. Diese führt die SAMF durch. Die Simulationsstudien umfassen zum einen den generellen Vergleich zwischen simultaner, sequenzieller und selbstorganisierter Optimierung der Ablaufplanung von Maschinen und FTS. Zum anderen wird der vorgestellte Ansatz mit anderen, in der vorangegangenen Literaturanalyse vorgestellten Verfahren verglichen. Die Reaktionsfähigkeit der Steuerungsarchitektur auf unerwartet auftretende Ereignisse wird ebenfalls untersucht.

4.1 Entwicklung des Algorithmus zur simultanen Ablaufplanung

Die Vorstellung des zur SAMF entwickelten Optimierungsalgorithmus sowie die Erläuterung der Neuplanungsstrategie erfolgt in diesem Unterkapitel. Zur Auswahl des Optimierungsverfahrens werden die einzelnen, in Kapitel 2.3 vorgestellten Verfahren für kombinatorische Optimierungsprobleme, gegenübergestellt (Tabelle 7) und bzgl. folgender Kriterien bewertet:

- *Optimierungsqualität*: Bezeichnet die Qualität des Ergebnisses, welches bzgl. der Optimierung nach einer vorgegebenen Zielfunktion erreicht werden kann.
- *Rechenzeit (RZ)*: Bezeichnet, wie lange der Algorithmus zur Erstellung einer Lösung des Problems sowie der Optimierung dieser benötigt.
- *Integrationsaufwand*: Bezeichnet den Aufwand, der benötigt wird, um das entsprechende Verfahren auf das vorliegende Problem zu applizieren.
- *Anpassungsaufwand*: Bezeichnet den Aufwand für Anpassung an dem jeweiligen, verfahrensbasierten Optimierungsalgorithmus, der benötigt wird, um Änderungen des vorliegenden Problems abbilden zu können. Beispielhaft hierfür sind geänderte Randbedingungen.

Die Vorgehensweise zur Bewertung dieser Kriterien ist Folgende: Das Verfahren, welches das jeweilige Bewertungskriterium am besten erfüllt wird mit „++“ bewertet und das Verfahren, das dieses am wenigsten erfüllt, mit „-“. Für die übrigen Verfahren wird anschließend entschieden, in welche Richtung sie tendieren. Wichtig ist hierbei anzumerken, dass es sich um eine Generalisierung der allgemein bekannten Vor- und Nachteile der einzelnen Verfahren handelt. Dieser Vergleich darf deshalb lediglich als Auswahlhilfe für das am wahrscheinlich geeignetsten Verfahren für den vorliegenden Einsatzfall angesehen werden.

Exakte Verfahren haben den Vorteil, dass sie das globale Optimum als Lösungen liefern können. Der Integrationsaufwand ist bei diesen Verfahren meist gering. Begründet ist dies dadurch, dass das Problem lediglich mathematisch beschrieben werden muss. Die anschließende Lösung des mathematischen Problems erfolgt meist durch kommerzielle, oder frei verfügbare Solver. Diese müssen nicht mehr auf das konkret vorliegende Problem angepasst werden. Die komplette mathematische Beschreibung komplexer Probleme kann allerdings ein Problem darstellen. Der große Nachteil exakter Verfahren ist die hohe RZ, welche sich durch die Forderung zum Finden des globalen Optimums ergibt. Aufgrund ihrer hohen RZ scheiden sie damit für die Steuerung eines RS aus.

Einfache Konstruktionsverfahren, wie bspw. *Prioritätsregeln*, liefern in geringer RZ einen gültigen Ablaufplan und sind einfach auf eine Problemstellung zu applizieren. Auch die Anpassung an geänderte Randbedingungen des vorliegenden Problems geht entsprechend schnell und einfach. Durch diese Vorteile sind Konstruktionsverfahren in kommerziellen Systemen weit verbreitet. Allerdings bieten diese Verfahren keine gute Optimierungsqualität, weshalb sie als geeignete Verfahren für den vorliegenden Anwendungsfall ausscheiden.

Eine Möglichkeit, die Lösung eines Konstruktionsverfahrens zu verbessern, ist es, basierend auf dessen Ergebnis ein Verbesserungsverfahren, wie bspw. *lokale Suche*, anzuwenden. Dadurch lässt sich eine Verbesserung der Optimierungsqualität erzielen, was im Gegenzug mit einer Verschlechterung in den Punkten RZ sowie Integrations- und Anpassungsaufwand einhergeht.

Durch Metaheuristische Verfahren kann eine weitere Verbesserung der Optimierungsqualität erreicht werden, die sich durch die Anwendung komplexerer und effizienterer Algorithmen ergibt. Die RZ ist abhängig von der verwendeten Metaheuristik sowie der vorgegebenen Abbruchbedingung. Großer Nachteil der Anwendung von diesen Verfahren ist der hohe Integrationsaufwand. Bereits kleine Änderungen an der Problemstellung können zu einem großen Anpassungsaufwand führen [179].

Gegenteilig zu dem hohen Integrations- und Anpassungsaufwand von metaheuristischen Verfahren, bietet die CP als Verfahren der KI, eine schnelle und einfache Integration und Adaption an das vorliegende Problem. Die Beschreibung des Problems erfolgt, ähnlich den exakten Verfahren, durch Randbedingungen (Constraints) zwischen den einzelnen Operationen, Aufträgen und Maschinen. Diese werden anschließend von einem CP-Solver gelöst, welcher nicht auf das spezifische Problem angepasst werden muss. Der Vorteil dieses Verfahrens gegenüber den exakten Verfahren ist die wesentlich geringere RZ. Die Optimierung kann nach einer vorgegebenen maximalen Zeit abgebrochen werden und die beste bis dahin gefundene, gültige Lösung verwendet werden. Die CP liefert in angemessener Zeit zwar unter Umständen schlechtere Ergebnisse als spezialisierte metaheuristische Algorithmen, diese Differenz fällt allerdings nur gering aus [180].

Optimierungsverfahren für kombinatorische Probleme							
Kriterien	Exakte Verfahren	Konstruktions-verfahren	Verbesserungs-verfahren	Metaheuristische Verfahren	KI Verfahren (CP)	Legende	
Optimierungsqualität	++	-	**0**	+	+	-	Schlecht
Rechenzeit	-	++	+	**0**	**0**	0	Mittel
Integrationsaufwand	+	+	**0**	-	++	+	Gut
Anpassungsaufwand	+	+	**0**	-	++	++	Sehr gut

Tabelle 7: Bewertung und Vergleich der verschiedenen Optimierungsalgorithmen für kombinatorische Probleme.

CP besitzt die höchste Übereinstimmung mit den gestellten Auswahlkriterien. Dieses wird deshalb zur Lösung und Optimierung der SAMF, innerhalb der Scheduler Komponente, genutzt.

4.1.1 CP-Modell zur SAMF ohne alternative Maschinen (JSSP-Umgebung)

Im Folgenden wird das entwickelte CP-Modell der SAMF vorgestellt. Die hierzu ausgewählte CP-Sprache ist das von *IBM* entwickelte Konzept *ILOG CPLEX CP Optimizer*. Die wichtigsten Grundelemente dieses Konzepts werden folgend vorgestellt [179]. Alternativ können auch andere CP Konzepte verwendet werden, wobei das grundlegend vorgestellte CP-Modell erhalten bleibt. Im Anhang J erfolgt eine detaillierte Erläuterung der Funktionsweise von *ILOG CPLEX CP Optimizer*.

Ein grundlegendes Element bei der CP ist die Intervallvariable. Eine Intervallvariable (***intervalVar()***) beschreibt ein Zeitintervall, in dem eine bestimmte Eigenschaft enthalten ist. Beispiel hierfür ist die Bearbeitung einer Operation, welche durch die entsprechende Prozesszeit repräsentiert ist. Wird ein Zeitintervall durch eine Dauer beschrieben, wird diesem im Rahmen der Lösungsfindung ein Startzeitpunkt sowie ein sich daraus ergebender Endzeitpunkt zugewiesen, oder umgekehrt.

Zeitintervalle können als optional definiert werden: ***intervalVar x, optional***. In diesem Fall ist es Teil des Lösungsprozesses, zu entscheiden, ob das Intervall in der Lösung vorhanden sein wird oder nicht. Liegt bspw. ein Transportauftrag vor, der alternativ von zwei FTS durchgeführt werden kann, erfolgt für das jeweilige FTS die Definition einer optionalen Intervallvariablen. Teil des Lösungsprozesses ist es zu entscheiden, welches FTS diesen Transport durchführt. Die Intervallvariable, welche die Transportfahrt des hierzu selektierten FTS beschreibt, ist in der Lösung des Ablaufplanungsproblems vorhanden. Die Intervallvariable des nicht zur Ausführung dieses Transportes selektieren FTS, ist entsprechend nicht vorhanden.

Liegen als optional definierte Intervallvariablen vor, ist eine ***alternative()*** Constraint notwendig. Diese definiert, welche optionalen Intervallvariablen bzgl. einer Selektion miteinander konkurrieren. Ist bei dem Constraint *alternative(x, [y_1,..., y_4])* die Intervallvariable *x* in der Lösung anwesend, ist auch genau eine der Intervallvariablen {y_1, ..., y_4} anwesend und mit *x* bzgl. dem Start- und Endwert synchronisiert. Ist *x* hingegen nicht anwesend, sind auch alle {y_1, ..., y_4} nicht anwesend. {y_1, ..., y_4} entsprechen optionalen Intervallvariablen und durch den *alternative()* Constraint wird definiert, welche alternativ möglichen Intervallvariablen vorliegen, wenn *x* anwesend ist. Äquivalent zu dem vorherigen Beispiel bzgl. zwei verfügbarer FTS zur Durchführung eines Transports, lässt sich auch die *alternative()* Constraint erläutern. Diese beschreibt durch *alternative(x, [y_1, y_2])*, dass die Transportfahrt *x* entweder von FTS 1 (y_1), oder FTS 2 (y_2) durchgeführt werden kann. y_1 und y_2 sind optionale Intervallvariablen welche die Transportfahrt durch das jeweilige FTS repräsentieren. Abbildung 29 stellt den Transport *x* und die beiden Transportmöglichkeiten durch FTS 1 (y_1) und FTS 2 (y_2) in Form von Blöcken dar. Die linke Seite zeigt die drei Intervallvariablen *x*, y_1, y_2 vor der Lösungsfindung respektive Optimierung und die rechte Seite das Ergebnis dieser. Der Transport *x* wird von FTS 2 durchgeführt. Aufgrund dessen ist y_2 in der Lösung vorhanden, y_1 hingegen nicht. Die Transportfahrt *x* und die Ausführung dieser durch FTS 2 (y_2) sind bzgl. der Dauer sowie der jeweiligen Start- und Endzeit identisch.

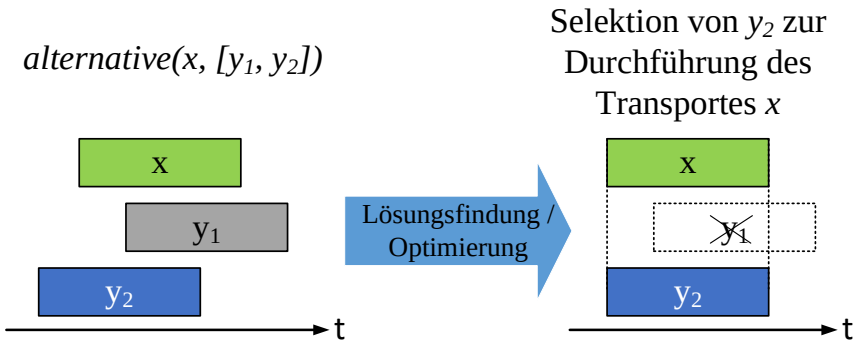


Abbildung 29: CP Constraint *alternative(x, [y_1, y_2])*

Ein Satz von mehreren Intervallvariablen kann durch eine ***intervalSequenceVar()*** (Intervallfolgevariable) zusammengefasst werden. Die Intervallfolgevariable repräsentiert die Gesamtreihenfolge der da-

rin enthaltenen Intervallvariablen. Es erfolgt keine Berücksichtigung nicht anwesender Intervallvariablen. Eine Intervallvariable ist nicht anwesend, wenn dies eine Verletzung der problembeschreibenden Randbedingungen verursachen würde. Bspw. können alle auf einer Maschine auszuführenden Operationen durch eine Intervallfolgevariable repräsentiert werden. Zur Verdeutlichung des Inhalts und des Wertes einer Intervallfolgevariablen soll folgendes Beispiel dienen:

Intervallfolgevariable *p* über die Intervallvariablen {*x1, …, x5*}:

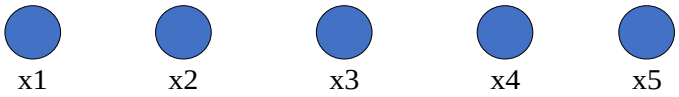


Abbildung 30: Intervallfolgevariable *p*

Der Wert der Intervallfolgevariable *p* entspricht einer Permutation der anwesenden Intervallvariablen:

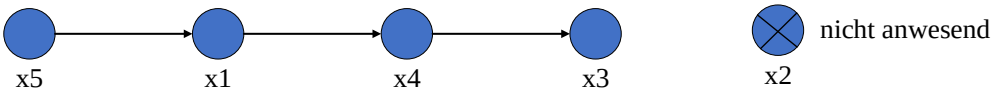


Abbildung 31: Wert der Intervallfolgevariable *p*

Die Reihenfolge der in der Permutation anwesenden Intervallvariablen impliziert allerdings keine zeitliche Ordnung. Eine zeitliche Ordnung ergibt sich durch die für die Intervallfolgevariable definierten Constraints.

endbeforeStart() ist ein solcher Constraint. Dieser definiert reihenfolgebedingte Randbedingungen, wie bspw. die Operationsreihenfolge zur Fertigung eines Produktes. Abbildung 32 soll dies am Beispiel *endBeforeStart(x_i, x_j, z_{ij})* verdeutlichen. x_i muss abgeschlossen sein bevor x_j, durchgeführt werden kann. Der Wert z_{ij} beschreibt eine Zeit, die zwischen dem Ende von x_i, also $e(x_i)$ und dem Start von x_j, also $s(x_j)$ vergehen muss. Hierdurch kann bspw. die Rüstzeit zwischen zwei aufeinanderfolgenden Bearbeitungen auf einer Maschine definiert und in der Lösungsfindung berücksichtigt werden.

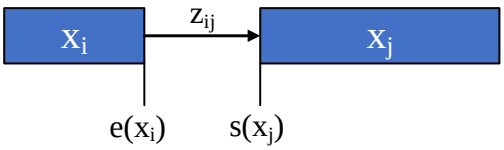


Abbildung 32: CP Constraint *endBeforeStart(x_i, x_j, z_{ij})*

Der ***noOverlap()*** Constraint dient zur Einhaltung ressourcenbedingter Randbedingungen. Beispielhaft hierfür ist die Definition, dass auf einer Ressource zu jedem Zeitpunkt nur eine Operation ausgeführt werden darf. Liegen die Intervallvariablen {x_1, …, x_5} vor und werden durch den Constraint *noOverlap([x_1, …, x_4])* restringiert, kann eine mögliche Lösung wie folgt aussehen:

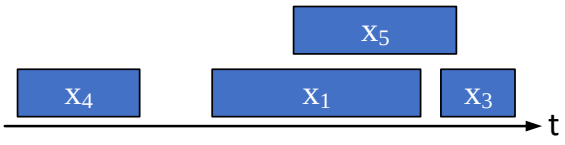


Abbildung 33: CP Constraint *noOverlap([x_1, …, x_4])*

Wie in Abbildung 33 zu sehen ist, gibt es bei den mittels *noOverlap()* Constraint restringierten Intervallvariablen keine zeitliche Überschneidung. Die Reihenfolge sowie die Anwesenheit (x_2 ist nicht anwesend) ist beliebig, da hierzu keine Constraints festgelegt wurden. x_5 ist nicht in der *noOverlap()* Constraint inkludiert und kann deshalb zeitlich parallel zu den übrigen Intervallvariablen sein. Häufig muss zwischen zwei aufeinanderfolgenden Operationen auf einer Ressource, bedingt durch Rüstzeiten einer Maschine, eine bestimmte minimale Zeit vergehen. Diese Zeiten sind oftmals von den beiden jeweiligen, aufeinanderfolgenden Operationen abhängig. Zum Modellieren solcher Randbedingungen kann die *noOverlap()* Constraint durch eine ***transition distance matrix*** (Übergangsdistanzmatrix) erweitert werden. Zur Verdeutlichung der Funktionsweise dieser, wird der Constraint *noOverlap([x_1, …, x_4], M, direct)* betrachtet. *M* ist die verwendete und in Abbildung 34 dargestellte Übergangsdistanzmatrix. Jeder Intervallvariablen $x_1, \ldots, x_4$ wird der Index *0, 1,* oder *2* zugeteilt und über den einzelnen, die Intervallvariablen repräsentierenden, Blöcken dargestellt. Der unter den Pfeilen eingetragene Wert gibt den minimalen, zeitlichen Abstand zwischen zwei aufeinander folgenden Intervallvariablen an. Dieser Wert entspricht dem Wert in der Übergangsdistanzmatrix, der sich in der Zeile des Index der Vorgängeroperation und der Spalte des Index der Nachfolgeoperation befindet. Der boolesche Parameter *direct* gibt an, ob der Übergangsabstand nur auf direkte (*direct = true*) oder auch auf indirekte Nachfolger (*direct = false, default*) angewendet wird.

Übergangsdistanzmatrix M

	0	1	2
0	0	40	50
1	20	5	45
2	30	25	0

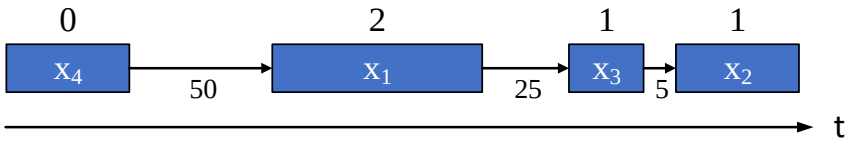


Abbildung 34: CP Constraint noOverlap() mit transition distance matrix

Zielfunktionen können auf Intervallvariablen basierend definiert werden. Ein Beispiel hierfür ist:

minimize max(i in 1..n) endOf(x[i])

x[i] ist eine Liste von Intervallvariablen, die durchzuführende Operationen repräsentieren. Als Optimierungskriterium werden durch *endOf(x[i])* allgemein die Endzeiten der Operationen und durch *max(i in 1..n)* genau die Endzeit der Operation optimiert, die von allen vorhandenen Operationen den spätesten Endzeitpunkt besitzt. Durch *minimize* wird angegeben, dass es sich bei der Optimierung um eine Minimierung handelt.

Im Folgenden wird das in dieser Arbeit entwickelt CP-Modell zur Beschreibung und Optimierung der SAMF vorgestellt. Dazu werden die zuvor erläuterten Elemente der CP verwendet.

Variablen:

M: Menge von Maschinen, $\{1, 2, \ldots, m\}$

P: Menge von Aufträgen, $\{1, 2, \ldots, n\}$

S_p: Menge von Operationen des Auftrags p, $\{1, 2, \ldots, k\}$, $p \in P$

F: Menge von FTS, $\{1, 2, \ldots, a\}$

A: Menge von Transportaufträgen, {1, 2, …, l}

b_{sp}: Bearbeitungszeit der Operation s des Auftrags p, $s \in S$, $p \in P$

t_{avw}: Transportzeit der Fahrt a von Maschine v nach Maschine w, $a \in A, v, w \in M$

e_{vw}: Leerfahrtzeit der Fahrt von Maschine v nach Maschine $w, v, w \in M$

L: Leerfahrtenmatrix (l x l Matrix)

c_{max}: DLZ

Intervallvariablen:

Intervallvariable für jede zu bearbeitende Operation. Der jeweilige Wert entspricht b_{sp}

interval s_{pi}

Intervallvariable für jeden durchzuführenden Transportauftrag. Der jeweilige Wert entspricht t_{avw}

interval a_j

Intervallvariable für jeden durchzuführenden Transportauftrag a_j für jedes vorhandene FTS f. Die Intervallvariable ist optional, da das jeweilige FTS den Transportauftrag nur ausführen muss, wenn kein anderes FTS diesen ausführt. Der jeweilige Wert (Transportzeit) entspricht t_{avw}.

interval a_{fj}, *optional*

Die Übergangsdistanzmatrix L dient der Definition von Leerfahrten zwischen zwei aufeinanderfolgenden Transportaufträgen eines FTS. Die jeweilige Übergangsdistanz entspricht der Fahrtzeit vom Endpunkt des vorherigen Transportauftrags zum Startpunkt des aktuellen Transportauftrags. Die Matrix L hat die Größe y x y, wobei y die Anzahl der durchzuführenden Transportaufträge ist.

transitionDistance L

Diese ist in Tabelle 8 exemplarisch dargestellt. Alle durchzuführenden Transportaufträge a_j sind gegeneinander aufgetragen. Die aktuellen Transportaufträge a_j, sind in den Spalten und deren mögliche Vorgänger a_{j-1} in den Zeilen aufgelistet. v entspricht dem Endpunkt und w dem Startpunkt des jeweiligen Transportauftrags. In der Diagonalen sind alle Einträge mit „-“ gekennzeichnet, da jeder Transportauftrag einmalig ausgeführt wird und ein Transportauftrag sich nicht selbst als Vorgänger haben kann. Die eingetragenen Werte entsprechen den Leerfahrtzeiten e_{vw} von Maschine v zu Maschine w. Eine Leerfahrt startet am Endpunkt der vorangegangenen Transportfahrt a_{j-1} und endet am Startpunkt der folgenden Transportfahrt a_j.

Die Matrix L wird den Intervallfolgevariablen F_j der einzelnen FTS mittels einer *noOverlap(F_j, L, true)* Constraint zugeordnet.

			a_j				
			a_1	a_2	a_3	…	a_y
		v \ w	3	1	2	w	4
a_{j-1}	a_1	4	-	10	8	e_{vw}	0
	a_2	3	0	-	6	e_{vw}	6
	a_3	1	8	0	-	e_{vw}	10
	…	v	e_{vw}	e_{vw}	e_{vw}	-	e_{vw}
	a_y	2	6	6	0	e_{vw}	-

Tabelle 8: Exemplarische Leerfahrtmatrix. Die aufgezeigten Fahrzeiten entsprechen den Fahrzeiten aus Layout 1 der BI von Bilge und Ulusoy in Abbildung 38.

Werden einem FTS die Transportaufträge a_1, a_3 und a_y zugeteilt, ergibt sich der in Abbildung 35 gezeigte Ablaufplan für dieses FTS. Die einzelnen Blöcke entsprechen den Intervallvariablen der auszuführenden Transportaufträge a_1, a_3 sowie a_y, wohingegen die Pfeile zwischen den Blöcken die notwendigen Leerfahrten repräsentieren. Die Dauer der Leerfahrten ist an den Pfeilen angebracht und entspricht den zugehörigen Werten aus Tabelle 8. Über den Blöcken sind jeweils die Maschine *w* (links), an welcher der Transportauftrag startet und die Maschine *v* (rechts), an der der Transportauftrag endet, aufgeführt.

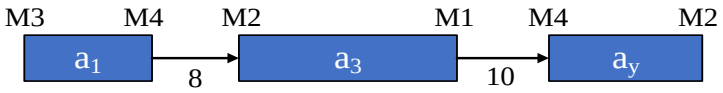


Abbildung 35: Anwendung Leerfahrtenmatrix für ein FTS

Intervallfolgevariablen:

Intervallfolgevariable für jede Maschine, welche die Intervallvariablen der jeweiligen auf der Maschine zu bearbeitenden Operationen umfasst

intervalSequenceVar(s_{pi}) M_i

Intervallfolgevariable für jedes FTS, welche die Intervallvariablen der jeweiligen, durch das FTS ausgeführten, Transportaufträge umfasst

intervalSequenceVar(a_{fj}) F_j

Randbedingungen (Constraints):

Einhaltung der Operationsreihenfolge für jeden Auftrag *p*

endBeforeStart(s_{pi-1}, s_{pi})

Einhaltung der Transportreihenfolge für jedes FTS *f*

endBeforeStart(a_{fi-1}, a_{fi})

Transport muss beendet sein, bevor die Bearbeitung auf der Maschine starten kann

endBeforeStart(a_{fi}, s_{fi})

Vorgängeroperation muss beendet sein, bevor Transport zur nächsten Maschine starten kann

endBeforeStart(s_{pi-1}, a_{fi})

Jeder Transportauftrag kann alternativ von jedem verfügbaren FTS ausgeführt werden

alternativ(a_{fi})

Eine Maschine kann zu jeder Zeit nur eine Operation bearbeiten

noOverlap(M_i)

Jedes FTS kann zu jedem Zeitpunkt nur einen Transportauftrag ausführen. Ein FTS muss, wenn die aktuelle Position des FTS und der Startpunkt des anliegenden Transportauftrags nicht identisch sind, noch eine entsprechende Leerfahrt zum Startpunkt durchführen. Die Fahrzeit dieser Leerfahrt ist in der Übergangsdistanzmatrix *L* enthalten.

noOverlap(F_j, L, true)

Zielfunktion:

Minimierung der DLZ. *s[i]* entspricht dabei einem Array, das alle Operationen eines Auftragssatzes beinhaltet.

minimize max(i in 1..n) endOf(s[i])

Ladevorgänge der FTS könnten innerhalb der Ablaufplanung ebenfalls berücksichtigt werden. In dem vorgestellten Ansatz erfolgt dies allerdings nicht, da es sich um eine Feinplanung mit kurzem Zeithorizont handelt. Dem gegenübergestellt ist die Ladedauer von mehreren Stunden sowie die Betriebszeit eines FTS von bis zu einem halben Tag [181]. Der Planungshorizont ist zu gering, dass die Berücksichtigung eines in der Ferne liegenden Ladevorgangs eine Änderung an dem erstellten Ablaufplan hervorrufen würde. Diese Berücksichtigung würde dennoch eine Steigerung der Komplexität des Ablaufplanungsproblems verursachen und dadurch eine Erhöhung der RZ mit sich ziehen. Die Transport Komponente des FTS trifft die Entscheidung über einen notwendigen Ladevorgang stattdessen dezentral selbst und teilt dies der Koordinator Komponente mit. Bei der anschließenden Neuplanung wird die Verfügbarkeit des betroffenen FTS auf den Zeitpunkt gelegt, an dem der Ladevorgang voraussichtlich abgeschlossen sein wird. Der Ablauf einer Neuplanung wird in Kapitel 4.1.3 genauer erläutert.

Das zuvor beschriebene CP-Modell wird in Java unter Nutzung der API von *IBM ILOG CPLEX Optimization Studio 12.10.0* implementiert. Zur Lösung des Problems wird der Solver *CP Optimizer* genutzt. *Large Neighborhood Search (LNS)* wird zur Optimierung des CP-Modells der SAMF verwendet. Eine ausführliche Beschreibung des genutzten *LNS*-Algorithmus kann in [182] gefunden werden. Die nachfolgend beschriebenen CP-Modelle werden ebenfalls unter Nutzung dieser Konstellation implementiert.

4.1.2 CP-Modell zur SAMF mit alternativen Maschinen (FJSSP-Umgebung)

Das zuvor vorgestellte CP-Modell ist auf eine JSSP-Umgebung beschränkt. In RS gibt es zur Bearbeitung einer Operation allerdings oftmals mehrere, ähnliche oder gar identische Maschinen. Um dies ebenfalls bei der Erstellung und Optimierung der SAMF berücksichtigen zu können, muss das zuvor

präsentierte CP-Modell auf eine FJSSP-Umgebung erweitert werden. Hierzu erfolgt eine Erweiterung der Variablen, der Intervallvariablen, eine Ergänzung um entsprechende, zusätzliche Randbedingungen sowie eine Anpassung der Leerfahrtenmatrix. Im Folgenden werden alle innerhalb der FJSSP-Umgebung zusätzlich notwendigen Elemente aufgeführt. Alle anderen im JSSP-Kontext zuvor gegebenen Elemente sind weiterhin gültig.

Innerhalb der JSSP-Umgebung ist jeder Operation eines Auftrags genau eine Maschine, die sie bearbeitet und damit eine sich ergebende Prozesszeit zugeordnet. In der FJSSP-Umgebung hingegen kann jede Operation von einem Satz alternativer Maschinen bearbeitet werden. In Abhängigkeit der zur Bearbeitung einer Operation ausgewählten Maschine, kann die notwendige Prozesszeit variieren.

O_{spm}: Menge von Maschinen aus M für die Bearbeitung der Operation s des Auftrags p, $\{1, 2, \ldots, o\}$, $s \in S$, $m \in M$, $p \in P$

b_{spm}: Bearbeitungszeit der Operation s des Auftrags p auf Maschine m, $s \in S$, $m \in M$, $p \in P$

Y: Leerfahrtenmatrix (y x y Matrix)

Intervallvariablen:

Für jede Bearbeitungsmöglichkeit einer jeden Operation muss eine eigene Intervallvariable definiert werden. Der jeweilige Wert entspricht b_{spm}. Diese Intervallvariable wird als optional definiert, da jede Operation nur von einer der für ihre Bearbeitung in Frage kommenden Maschinen tatsächlich bearbeitet wird.

interval o_{spmi}*, optional*

Eine Intervallvariable muss für jeden möglichen Transportauftrag definiert werden. Die Anzahl der Transportaufträge hängt von den Bearbeitungsmöglichkeiten der aktuellen sowie der nachfolgenden Operation ab. Gibt es für die Bearbeitung der aktuellen und der nachfolgenden Operation eines Auftrags bspw. jeweils drei mögliche Maschinen, ergeben sich bis zu neun mögliche Transportaufträge. Die Dauer des jeweiligen Transports entspricht t_{avw}. Diese Intervallvariablen werden als optional deklariert, da für jedes Paar an Operation und Nachfolgeroperation nur einer der möglichen Transportaufträge ausgeführt wird. In der Lösung ist somit nur der tatsächlich auszuführende Transportauftrag enthalten. Wenn die aktuelle Operation auf der gleichen Maschine wie die Nachfolgeoperation bearbeitet wird, ist kein Materialtransport notwendig, und es wird entsprechend keine Intervallvariable angelegt.

interval $a_{jo,\, o+1}$*, optional*

Die Übergangsdistanzmatrix Y definiert die Dauer der Leerfahrten zwischen zwei aufeinanderfolgenden Transportaufträgen eines FTS. Die jeweilige Übergangsdistanz entspricht der Fahrtzeit vom Endpunkt des vorherigen Transportauftrags zum Startpunkt des aktuellen Transportauftrags. Y ist eine Matrix der Größe y x y. y entspricht der Anzahl aller möglichen Transportreihenfolgen, zwischen den, zur Bearbeitung der Vorgängeroperation und der aktuellen Operation verfügbaren Maschinen.

transitionDistance Y

Randbedingungen (Constraints):

Jede Operation kann alternativ von einer der für ihre Bearbeitung verfügbaren Maschinen bearbeitet werden

alternativ(o_{spm})

Alle weiteren Elemente der CP-Definition der JSSP-Umgebung bleiben auch in der FJSSP-Umgebung erhalten und werden lediglich bzgl. der neuen Indizierung angepasst.

In diesem und dem vorherigen Kapitel wurden die CP-Modelle zur SAMF in deterministischen Umgebungen vorgestellt. Diese sind nicht in der Lage auf unerwartete eintreten Ereignisse zu reagieren, wodurch es in einem solchen Fall zu einem Stillstand des RS kommen kann. Eine Erweiterung der CP-Modelle sowie die Kooperation der Komponenten der Steuerungsarchitektur untereinander ist notwendig, um adäquat auf solche Ereignisse reagieren zu können. Ziel dabei ist die Erstellung eines neuen, gültigen Ablaufplans. Die notwendigen Erweiterungen sowie die Kooperation der Komponenten während einer Neuplanung, werden im folgenden Kapitel dargestellt und erläutert.

4.1.3 Neuplanung beim Auftreten unerwarteter Ereignisse

In RS treten häufig unerwartet Ereignisse auf. Beispiele hierfür sind:

- Ressourcenausfälle
- Ausfall eines FTS
- Notwendiger Ladevorgang eines FTS
- Verzögerungen während der Ausführung von Bearbeitungsprozessen
- Neue zu bearbeitende Aufträge
- Änderungen der notwendigen Refabrikationsprozessschritte nach einem Inspektionsprozess

Diese führen zu Verzögerungen des Ablaufplans, oder gar dazu, dass dieser undurchführbar wird. Der Stillstand des RS wäre die Folge. Das Steuerungssystem muss einen neuen, gültigen Ablaufplan erstellen, wenn ein solches Ereignis eintritt. Der aktuelle Status sowie die Verfügbarkeit aller relevanten Fertigungsteilnehmer muss hierbei berücksichtigt werden. Eine Neuplanung kann auch sinnvoll sein, wenn der Ablaufplan zwar noch gültig, aber stark verzögert ist. Der durch die Verzögerungen nicht mehr optimale Ablaufplan kann im Rahmen der Neuplanung an die aktuelle Situation angepasst und bzgl. dieser neu optimiert werden. Zur Neuplanung müssen folgende Informationen in aktueller Form vorliegen:

- Anzahl der aktuell im Umlauf befindlichen Produkte/Aufträge
- Position der aktuell im Umlauf befindlichen Produkte/Aufträge innerhalb des Systems
- Status der einzelnen Operationen aller aktuell im Umlauf befindlichen Produkte/Aufträge
- In Kürze ins RS eintretende Produkte/Aufträge mit den entsprechenden Operationen und der Operationsreihenfolge
- Status und Verfügbarkeit der einzelnen Ressourcen innerhalb des RS
- Status und Verfügbarkeit der einzelnen FTS innerhalb des RS

Der Status einer Ressource oder eines FTS gibt an, ob das jeweilige Objekt defekt, verfügbar, oder aktuell in Bearbeitung befindlich ist. Führt eine Ressource oder ein FTS aktuell einen Prozess aus, gibt

die Verfügbarkeit an, zu welchem Zeitpunkt dieser abgeschlossen ist. Das Zustandsmodell bzgl. des Status der im Umlauf befindlichen Produkte wurde bereits in Kapitel 3.2.4 beschrieben.

Die Scheduler Komponente kann die Neuplanung erst starten, wenn all diese Informationen vorliegen. Diese Daten müssen vollständig und in aktueller Form vorliegend. Der neu erstellte Ablaufplan kann andernfalls, aufgrund unvollständiger, oder veralteter Annahmen, ungültig sein.

Die folgenden Komponenten kommunizieren während der Neuplanung miteinander:

- Koordinator Komponente
- Scheduler Komponente
- Ressourcen Komponenten
- Experten Komponenten
- Produkt Komponenten
- Transport Komponenten

Der Ablauf einer Neuplanung wird folgend am Beispiel eines Ressourcenausfalls dargestellt. Die Ressourcen Komponente der ausgefallenen Ressource übermittelt diesen an die Koordinator Komponente. Diese initialisiert und koordiniert die Neuplanung. Der chronologische Ablauf einer Neuplanung ist in Abbildung 36 dargestellt. Dieser Ablauf muss eingehalten werden, um sicher zu stellen, dass alle notwendigen Informationen über den aktuellen Status des RS vorliegen. Die Koordinator Komponente startet die Neuplanung, indem sie die Scheduler Komponente zur Durchführung einer neuen SAMF auffordert. Diese wartet anschließend mit der Formulierung des vorliegenden Ablaufplanungsproblems solange, bis sie die benötigten Informationen aller Transport -, Ressourcen -, Experten - und Produkt Komponenten erhalten hat. Die Scheduler Komponente formuliert das Ablaufplanungsproblem anschließend unter Einbeziehung des aktuellen Status des RS, und beginnt mit der SAMF. Der Zeitpunkt, an dem das System eine neue, vom Ablaufplan abhängige, Anweisung benötigt, dient als Abbruchkriterium der Optimierung. Ist zum Zeitpunkt der Initialisierung der Neuplanung bspw. ein FTS mit dem Transport eines Produktes beschäftigt, dient der Zeitpunkt, an dem dieser Transportvorgang abgeschlossen ist, als Zeitpunkt zum Abbruch der Optimierung. Der bis zu diesem Zeitpunkt beste gefundene Ablaufplan wird verwendet. Die Scheduler Komponente sendet nach der Fertigstellung der Optimierung die betreffenden Teile des Ablaufplans an die einzelnen Produkt -, Ressourcen - und Transport Komponenten. Abschließend wird mit der Ausführung des neuen Ablaufplans begonnen.

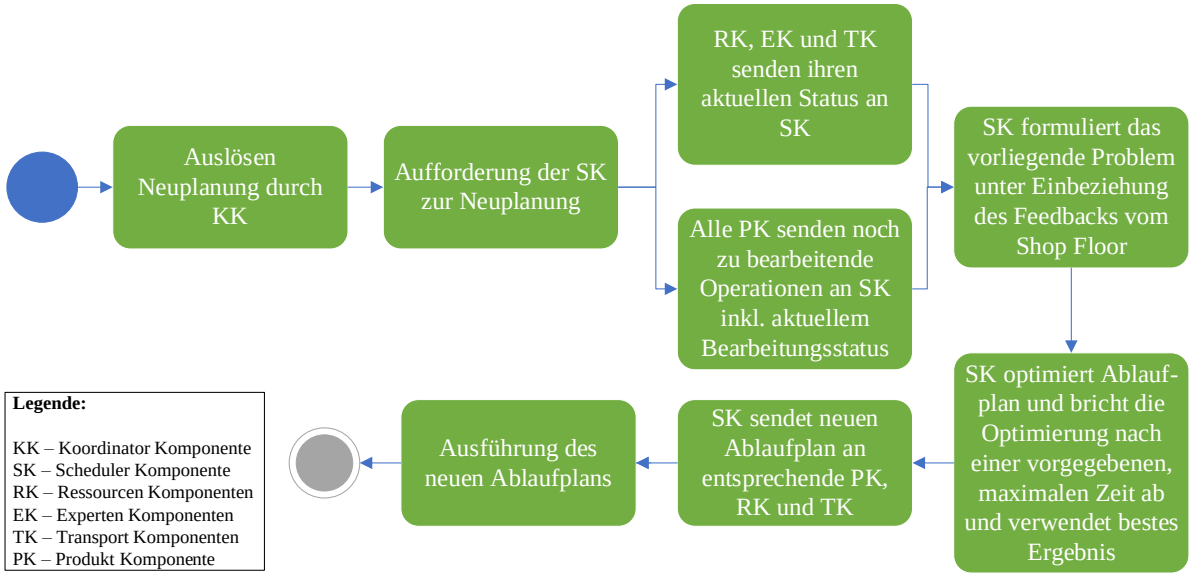


Abbildung 36: Chronologischer Ablaufplan zur Durchführung einer Neuplanung.

Die Scheduler Komponente verwendet zur Neuplanung eine modifizierte Variante des in Kapitel 4.1.1 vorgestellten CP-Modells zur SAMF. Die einzelnen Operationen, Ressourcen und FTS werden mit weiteren Randbedingungen bzgl. ihres Status und ihrer Verfügbarkeit versehen. Den Intervallvariablen der Operationen s_{pi}, sowie allen Intervallvariablen in den Intervallfolgevariablen der Maschinen $M_i(s_{pi})$ und denen der FTS $F_j(a_j)$ werden hierzu die Randbedingung *setStartMin(t)* zugeordnet. Die Variable *t* definiert den frühestmöglichen Startzeitpunkt für die entsprechende Intervallvariable. Ein Überblick, welchem Wert *t* bei den einzelnen Intervallvariablen entspricht, ist folgend gegeben:

- Operationen s_{pi}: .setStartMin(*Zeit, ab wann Operation beginnen kann*)
- Maschinen $M_i(s_{pi})$: .setStartMin(*Zeit, ab wann Ressource aktuelle Bearbeitung beendet hat*)
- Transport a_j: .setStartMin(*Zeit, ab wann Transportfahrt beginnen kann*)
- FTS $F_j(a_j)$: .setStartMin(*Zeit, ab wann FTS aktuellen Fahrauftrag beendet hat*)

Die im Folgenden dargestellten Code-Beispiele dienen der Verdeutlichung, wie die jeweiligen Zeitpunkte bestimmt werden. Hierbei ist zu beachten, dass unter *Erste Operation des Auftrags* die erste Operation der noch zu bearbeitenden Operationen zu versehen ist. Eine Aufführung bzgl. der Verfügbarkeit einer Maschine erfolgt nicht, da diese lediglich vom Endzeitpunkt der aktuell in Bearbeitung befindlichen Operation abhängig ist. Eine Maschine ist sofort verfügbar, wenn sich auf ihr zum aktuellen Zeitpunkt keine Operation in Bearbeitung, oder bereits an einem Pufferplatz befindet.

Bestimmung frühestmögliche Startzeit einer Operation:

Nur die erste Operation eines jeden Auftrags wird mit einer Verfügbarkeit versehen. Aufgrund dessen, dass alle weiteren Operationen eines Auftrags durch Reihenfolgenrandbedingungen nach dieser Operation stattfinden, erhalten diese automatisch den entsprechend frühestmöglichen Startzeitpunkt.

```
1. Wenn (Aktuelle Operation == Erste Operation des Auftrags) {
2.   Wenn (Verfügbarkeit Bearbeitungsmaschine > Endzeit Vorgängeroperation Auftrag) {
3.           Operation.setStartMin(Verfügbarkeit Bearbeitungsmaschine);
4.   }
5.   Sonst {
6.           Operation.setStartMin(Endzeit Vorgängeroperation Auftrag);
7.   }
8. }
```

```
9.  Sonst {
10.   Operation.setStartMin(Verfügbarkeit Bearbeitungsmaschine);
11. }
```

Bestimmung Verfügbarkeit FTS:

Die Verfügbarkeit eines FTS hängt von zwei Faktoren ab:

- Endzeitpunkt der in Ausführung befindlichen Transportfahrt
- Zeit für Leerfahrt zwischen aktueller Position des FTS und Startpunkt der nächsten beladenen Fahrt

```
1.  Für erste Operation eines jeden Auftrags
2.    Für jedes verfügbare FTS
3.            Wenn (Aktuelle FTS Position != Aktuelle Position Auftrag) {
4.                    Verfügbarkeit FTS = Verfügbarkeit FTS + Leerfahrtszeit;
5.                    FTS.setStartMin(Verfügbarkeit FTS);
6.            }
7.            Sonst {
8.                    FTS.setStartMin(Verfügbarkeit FTS);
9.            }
10.   }
11. }
```

Bestimmung frühestmögliche Startzeit eines Transportes:

Der Startzeitpunkt des durchzuführenden Transportes entspricht nicht zwangsläufig der Verfügbarkeit eines FTS. Ein FTS kann einen Auftrag erst von der aktuellen Maschine zur nächsten transportieren, wenn die in Bearbeitung befindliche Operation abgeschlossen ist. Der Startzeitpunkt einer Transportfahrt ist dadurch sowohl von der Verfügbarkeit des FTS, als auch vom Endzeitpunkt der Vorgängeroperation des zu transportierenden Auftrags abhängig.

```
1.  Für alle verfügbaren FTS {
2.            Wenn (Aktuelle Operation = Erste Operation des Auftrags) {
3.                    Wenn (Verfügbarkeit FTS > Endzeit Vorgängeroperation) {
4.                            Transport.setStartMin(Verfügbarkeit FTS);
5.                    }
6.                    Sonst {
7.                            Transport.setStartMin(Endzeit Vorgängeroperation);
8.                    }
9.            }
10.           Sonst {
11.                   Transport.setStartMin(Verfügbarkeit FTS);
12.           }
13. }
```

Der weitere Ablauf einer Neuplanung entspricht dem der Optimierung zur Erstellung eines initialen Ablaufplans. Alle Statusänderungen die sich voraussichtlich während Ausführung der Neuplanung ergeben werden, werden in dieser als Annahmen bereits berücksichtigt. Die Neuplanung müsste andernfalls bei jedem Ereignis unterbrochen und neu ausgeführt werden. Die Zustandsmodelle der einzelnen Komponenten werden allerdings erst dann aktualisiert, wenn die Ereignisse tatsächlich eingetreten sind. Dadurch stellen die Komponenten immer ein reales und aktuelles Abbild der Fertigungsebene dar. Im Rahmen der nächsten Neuplanung werden hierdurch zudem die tatsächlichen Statusänderungen und nicht die Annahme dieser berücksichtigt. Liegt eine Differenz zwischen angenommenen und tatsächlichen Statusänderungen vor, löst dies eine Neuplanung aus in der die tatsächlichen Statusänderungen berücksichtigt werden.

Nach der Darstellung der CP-Modelle zur SAMF sowie der Neuplanungsstrategie beim Auftreten unerwarteter Ereignisse, werden folgend die zur Verifizierung genutzten Simulationsumgebungen vorgestellt dargestellt.

4.2 Simulationsumgebung

Als Simulationsumgebungen dienen sowohl für die Untersuchungen der SAMF in einer JSSP - als auch einer FJSSP-Umgebung, in der einschlägigen Literatur bekannte Benchmarks. Die Verwendung von Benchmarks erlaubt es, den vorgestellten Ansatz mit anderen publizierten SotA-Ansätzen zu vergleichen.

4.2.1 Benchmarkinstanzen innerhalb einer JSSP-Umgebung

Die von Bilge und Ulusoy entwickelten BI [158] werden zur Untersuchung der SAMF in einer JSSP-Umgebung verwendet. Diese bestehen aus vier Maschinen, für deren räumliche Anordnung vier verschiedene Layoutvarianten vorhanden sind. Für jede der Layoutvarianten ergeben sich unterschiedliche Fahrzeiten zwischen den einzelnen Maschinen (siehe Abbildung 38 respektive Tabelle 9). Die BI beinhalten zudem zehn Auftragssätze, die in Abbildung 37 ausschnittsweise zu sehen sind. *M1* bis *M4* geben an, auf welcher Maschine die jeweilige Operation bearbeitet wird. Der sich hinter der jeweiligen Maschine in Klammern befindliche Wert entspricht der Bearbeitungszeit der Operation auf dieser Maschine. Die Auftragsätze bestehen jeweils aus vier bis acht Aufträgen und insgesamt 13 bis 21 zugehörigen Operationen. Diese müssen auf den vier Maschinen bearbeitet und durch zwei FTS zwischen den Maschinen transportiert werden. Die Kombination der vier Layoutvarianten mit den zehn Auftragssätzen ergibt insgesamt 40 Testinstanzen. Diese Testinstanzen besitzen alle ein *t/p*-Verhältnis (Transportzeit - /Prozesszeit - Verhältnis) > *0,25*. Das *t/p*-Verhältnis ist ein Indikator dafür, wie stark der Materialtransport die Ablaufplanung bzgl. der Maschinenbelegung beeinflusst [158]. Die vollständigen Auftragssätze sowie die Transportzeitenmatrizen sind im Anhang L zu finden.

Job Set 1	**Job Set 2**
Job 1: M1(8); M2(16); M4(12)	Job 1: M1(10); M4(18)
Job 2: M1(20); M3(10); M2(18)	Job 2: M2(10); M4(18)
Job 3: M3(12); M4(8); M1(15)	Job 3: M1(10); M3(20)
Job 4: M4(14); M2(18)	Job 4: M2(10); M3(15); M4(12)
Job 5: M3(10); M1(15)	Job 5: M1(10); M2(15); M4(12)
	Job 6: M1(10); M2(15); M4(12)

Abbildung 37: Ausschnitt der Daten bzgl. der als Testproblem genutzten Auftragssätze (Job Set‘s) für *t/p* > 0,25 [149].

Bilge und Ulusoy erzeugen, zur Betrachtung weiterer *t/p*-Verhältnisse, in ihren BI einen zusätzlichen Satz an Instanzen. Die Fahrzeiten werden hierzu halbiert und die Prozesszeiten mit einem Faktor von zwei respektive drei multipliziert. Dadurch ergeben sich 42 weitere Testinstanzen. Diese besitzen alle ein *t/p*-Verhältnis < *0,25*. Eine Aufführung der so erhaltenen Prozess- und Transportzeiten erfolgt nicht.

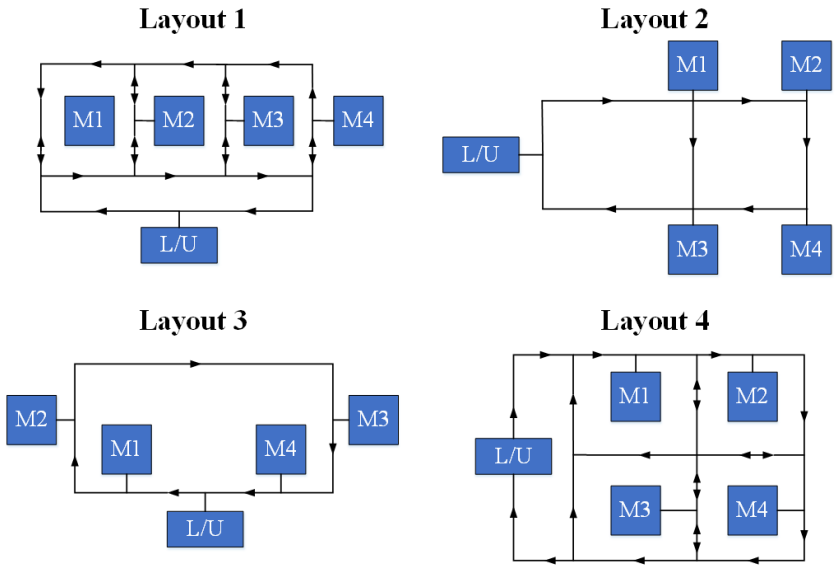


Abbildung 38: Layoutvarianten nach dem Testproblem von Bilge und Ulusoy [158].

Die Fahrzeiten zwischen den einzelnen Maschinen werden in Form einer Transportzeitenmatrix dargestellt (siehe Tabelle 9 für Layoutvariante 1). Diese Transportzeitenmatrix ist so zu lesen, dass die Transportzeit von einer Maschine in der linken Spalte zu einer Maschine in der oberen Zeile, dem Wert entspricht, an dem sich die jeweilige Zeile und Spalte kreuzen. Hierbei handelte es sich um eine einheitslose Zeiteinheit.

Layout 1	L/U	M1	M2	M3	M4
L/U	0	6	8	10	12
M1	12	0	6	8	10
M2	10	6	0	6	8
M3	8	8	6	0	6
M4	6	10	8	6	0

Tabelle 9: Transportzeitenmatrix für das Testproblem nach Bilge und Ulusoy für $t/p > 0{,}25$ [158].

Die folgenden Annahmen und Randbedingung gelten innerhalb des Benchmarks:

- FTS transportieren jeweils nur ein Produkt pro Fahrt.
- Notwendige Rüstzeiten sind in den Prozesszeiten inkludiert.
- Es sind zwei identische FTS für den Materialtransport vorhanden.
- Jedes Produkt startet und endet an der L/U Station (Load/Unload).
- Bearbeitungs-, Rüst-, Be-, bzw. Entladezeiten sind deterministisch.
- An jeder Maschine ist eine unbegrenzte Anzahl an Pufferplätzen vorhanden.
- Jede Maschine kann zu jedem Zeitpunkt höchstens eine Operation bearbeiten.
- Beide FTS sind in Bezug auf Geschwindigkeit und Leistungsfähigkeit identisch.
- Die FTS sind zuverlässig und frei von jeglichen Störungen während ihres Betriebs.
- Operationen können nach dem Start ihrer Bearbeitung nicht mehr unterbrochen werden.
- Jeder Auftrag hat eine vorgegebene Reihenfolge bzgl. seiner zu bearbeitenden Operationen.
- Das Ablegen und Aufnehmen der Produkte an den Maschinen wird nur von den FTS durchgeführt.
- Probleme, wie Verkehrsregelung, Stau, Maschinenausfall oder Ausfallzeiten, Ausschuss, Nacharbeit und Fahrzeugdisposition für Batterieladung, werden nicht berücksichtigt.

Die DLZ soll als Zielfunktion minimiert werden. Diese lässt sich wie folgt berechnen:

$T_{i,j}$ – Transportzeit der *j*-ten Operation des *i*-ten Auftrags

$P_{i,j}$ – Bearbeitungszeit der *j*-ten Operation des *i*-ten Auftrags

$L_{i,j}$ – Liege- und Wartezeit der *j*-ten Operation des *i*-ten Auftrags

DLZ einer Operation $O_{i,j} = T_{i,j} + P_{i,j} + L_{i,j}$

DLZ eines Auftrags $C_i = \sum_{i=1}^{n} O_{i,j}$

DLZ aller Aufträge (Makespan) = $\text{Max}(C_1, C_2, C_3, ..., C_m)$

4.2.2 Benchmarkinstanzen innerhalb einer FJSSP-Umgebung

Die BI von Kumar et al. [167] werden als FJSSP-Simulationsumgebung genutzt. Diese basieren auf den BI von Bilge und Ulusoy [158] und erweitern diese von einer JSSP- auf eine FJSSP-Umgebung. Sowohl die vier Layoutvarianten, als auch die dazugehörigen Transportzeitenmatrizen, bleiben unverändert. Die Auftragssätze sind so erweitert, dass jede Operation von insgesamt drei alternativen Maschinen bearbeitet werden kann. Der Auftragssatz 1 (*Job Set 1*) dieser BI ist als Beispiel hierfür in Abbildung 39 dargestellt. Bei Betrachtung der jeweils ersten Zeile von *Job1, Job2, …, Job5* ist festzustellen, dass die darin enthaltenen Maschinen und Bearbeitungszeiten der einzelnen Operationen, denen des ersten Auftragssatzes der BI von Bilge und Ulusoy (siehe Abbildung 37) entsprechen. Die einzelnen Aufträge sind jeweils um die Zeilen *Alt1* und *Alt2* erweitert. Diese beinhalten die beiden jeweiligen alternativen Maschinen, mit den zugehörigen Bearbeitungszeiten. Die allgemeine Problemstellung sowie die vorliegenden Randbedingungen entsprechen ebenfalls denen des Benchmarks von Bilge und Ulusoy. Alle zehn Auftragssätze werden hier aus Gründen der Übersichtlichkeit nicht präsentiert, sind allerdings im Anhang N zu finden.

Job Set 1

Job1: M1(8); M2(16); M4(12)
Alt1: M2(9); M3(14); M1(13)
Alt2: M3(9); M4(17); M2(10)
Job2: M1(20); M3(10); M2(18)
Alt1: M3(18); M1(13); M4(17)
Alt2: M2(21); M4(8); M3(19)
Job3: M3(12); M4(8); M1(15)
Alt1: M4(11); M2(10); M3(14)
Alt2: M1(11); M3(7); M2(17)

Job4: M4(14); M2(18)
Alt1: M1(16); M3(16)
Alt2: M3(16); M1(16)
Job5: M3(10); M1(15)
Alt1: M2(9); M4(16)
Alt2: M4(11); M3(14)

Abbildung 39: Ausschnitt der BI zur SAMF innerhalb einer FJSSP-Umgebung von Kumar et al. [167]

Nach der Vorstellung der genutzten Simulationsumgebungen sowie den verwendeten BI erfolgt anschließend die Erläuterung der verschiedenen Untersuchungen und die Präsentation der jeweiligen Simulationsergebnisse. In Kapitel 4.3 werden zuerst die Simulationsstudien innerhalb deterministischer und in Kapitel 4.4 anschließend in nicht deterministischer Umgebung dargestellt.

4.3 Simulationsstudien innerhalb deterministischer Umgebung

In diesem Kapitel wird das vorgestellte Optimierungsverfahren zur SAMF verschiedenen Simulationsstudien innerhalb einer deterministischen Umgebung unterzogen.

4.3.1 Vergleich verschiedener Metaheuristiken zur Lösung von CP-Modellen

Nachdem ein CP-Modell von einem Solver interpretiert wurde, erfolgt die Lösung und Optimierung des modellierten Problems. Metaheuristische Verfahren kommen bei kombinatorischen Optimierungsproblemen hierzu häufig zum Einsatz. Diese lassen sich, wie bereits erläutert, in einzellösungs- und mehrpopulationsbasierte Metaheuristiken untergliedern. LNS als einzellösungs- und GA als mehrpopulationsbasierte Metaheuristik finden für Ablaufplanungsprobleme häufig Anwendung. Die beiden Verfahren werden, zur Entscheidungsfindung welches der beiden im weiteren Verlauf Anwendung finden soll, gegenübergestellt.

Beide Verfahrensarten werden unter Nutzung der Java API von *IBM ILOG CPLEX Optimization Studio 12.10.0* implementiert. Die Simulationsstudie wird sowohl unter Nutzung einer JSSP-, als auch einer FJSSP-Umgebung durchgeführt. Alle BI mit einem *t/p-Verhältnis* > 0,25 werden bei der JSSP-Umgebung betrachtet. Somit werden insgesamt 80 BI untersucht. Für jede BI werden sowohl mit LNS als auch mit GA als Optimierungsverfahren, zehn Durchläufe vollzogen und der Mittelwert dieser zum Vergleich herangezogen. Die Optimierung wird beim Erreichen der besten bekannten Lösung für die entsprechende BI, oder alternativ nach 600 Sekunden RZ abgebrochen. Im zweiten Fall wird die bis dahin beste gefundene Lösung als Ergebnis gewertet. In Abbildung 40 sind die Durchschnittswerte über jeweils alle 40 BI innerhalb einer JSPP- und einer FJSSP-Umgebung für die DLZ und die benötigte RZ gegeben. Für die RZ wird zudem der Median angegeben, um die Abhängigkeit der Ergebnisse von einzelnen Ausreißern zu relativieren.

Die LNS liefert bzgl. DLZ und RZ sowohl in einer JSSP- als auch einer FJSSP-Umgebung gleiche, oder bessere Ergebnisse als der GA. Die Differenz zwischen LNS und GA fällt in einer FJSSP-Umgebung größer aus als in einer JSSP-Umgebung. Ein Grund hierfür ist der mit zunehmender Problemgröße bzw. –komplexität exponentiell wachsende Lösungsraum von Ablaufplanungsproblemen. Dadurch steigt die Differenz bzgl. der RZ zwischen LNS und GA von einer JSSP- auf eine FJSSP-Umgebung nicht linear, sondern exponentiell an. Die ausführlichen Ergebnisse für alle BI sind im Anhang Q in Tabelle 23 zu finden.

	JSSP-Umgebung				**FJSSP-Umgebung**			
	LNS		**GA**		**LNS**		**GA**	
	DLZ	**RZ**	**DLZ**	**RZ**	**DLZ**	**RZ**	**DLZ**	**RZ**
Ø	**108.4**	**11.49**	**108.4**	18.45	**74.3**	**3.87**	76.8	275.36
Median		**0.36**		0.39		**1.26**		102.89

Abbildung 40: Vergleich zwischen LNS und GA zur Lösung von CP-Modellen in JSSP- und FJSSP-Umgebung. Der jeweils beste Wert ist durch Fettschrift hervorgehoben.

Aufgrund der besseren Ergebnisse der LNS gegenüber dem GA wird für alle folgenden Untersuchung ebenfalls LNS als Optimierungsverfahren innerhalb der CP verwendet.

4.3.2 Vergleich simultane, sequenzielle und selbstorganisierte, simultane Ablaufplanung

Zur Verifikation einer möglichen DLZ-Reduktion durch SAMF, wird dieser Ansatz mit der sequenziellen sowie selbstorganisierten Ablaufplanung verglichen. Im ersten Teil erfolgt dieser Vergleich der drei Verfahren innerhalb einer JSSP-Umgebung und im Anschluss innerhalb einer FJSSP-Umgebung.

Vergleich der Verfahren innerhalb einer JSSP-Umgebung

Die BI von Bilge und Ulusoy werden zum Vergleich der Verfahren in einer JSSP-Umgebung genutzt. Sowohl das CP-Modell der SAMF, als auch das CP-Modell des JSSP innerhalb der sequenziellen Ablaufplanung werden in Java implementiert und mit dem Solver *CP Optimizer* von IBM gelöst. Dadurch soll ein vom Optimierungsalgorithmus möglichste unabhängiger Vergleich der zwischen simultanen und sequenziellen Ablaufplanung gewährleistet werden. Das in Kapitel 4.1.1 beschriebene CP-Modelle wird für die SAMF verwendet. Das CP-Modell des JSSP ist im Anhang O zu finden. Die Verwendung des Solvers *CP Optimizer* ermöglicht es, für beide Varianten das jeweilige globale Optimum, inklusive entsprechendem Nachweis, zu finden. Bei der sequenziellen Ablaufplanung erfolgt zuerst die Optimierung der DLZ für das JSSP. Die Zuordnung der FTS zu den sich ergebenden Transportaufträgen wird anschließend, basierend auf der besten gefundenen Lösung des JSSP, durchgeführt. Diese Zuordnung erfolgt durch die in [167] vorgestellte *Vehicle Assignment Heuristic (VAH)*. Die VAH besteht aus folgenden Prozessschritten:

1. Identifikation der aktuellen Position des FTS sowie der Zeit, zu der das FTS wieder verfügbar ist (VRT).
2. Berechnung der Fahrzeit von der aktuellen Position des FTS bis zur Maschine, an der sich das zu transportierende Produkt aktuell befindet.
3. Addition der Fahrzeit aus 2. zur VRT, um die Fertigstellungszeitpunkt der Leerfahrt (VET) zu berechnen.
4. Überprüfung, ob die aktuelle Operation des Produktes abgeschlossen ist oder nicht. Das FTS wartet mit dem Aufladen des Produktes bis die aktuelle Operation abgeschlossen ist, sollte dies bei Ankunft des FTS an der Maschine nicht bereits der Fall sein.
5. Vergleichen des Fertigstellungszeitpunkts der Operation und der VET. Der Spätere dieser beiden Zeitenpunkte wird für die weiteren Berechnungen verwendet.
6. Berechnung der Fahrzeit zwischen der Maschine, auf der sich der Auftrag aktuell befindet und der Maschine, auf der die Nachfolgeoperation des Auftrags ausgeführt wird.
7. Addition der Fahrzeit zu dem in Schritt 5 erhaltenen Wert. Hierdurch ergibt sich die Endzeit der beladenen Fahrt (VLT).

Dieser Prozess wird für jedes vorhandene FTS durchgeführt und die jeweils zugehörige VLT berechnet. Die VAH wählt das FTS mit der niedrigsten VLT für die Durchführung des zu vergebenden Transportauftrags aus. Das ausgewählte FTS ist für die nächste Zuordnung einer Fahrt bereit, sobald die zugeordnete Transportfahrt abgeschlossen ist. Die VRT der nächsten Fahrt entspricht der VLT der aktuellen Fahrt des FTS. Die Implementierung der VAH erfolgt ebenfalls in Java. Zur Durchführung der Zuordnung der FTS zu den Transportaufträgen, wird die durch den *CP Optimizer* Solver gefundene Lösung des JSSP über die Java API von *IBM ILOG CPLEX Optimization Studio 12.10.0* in Java importiert.

Die Vorgehensweisen der simultanen und der sequenziellen Ablaufplanung von Maschinen und FTS sind folgend als Ablaufdiagramme dargestellt.

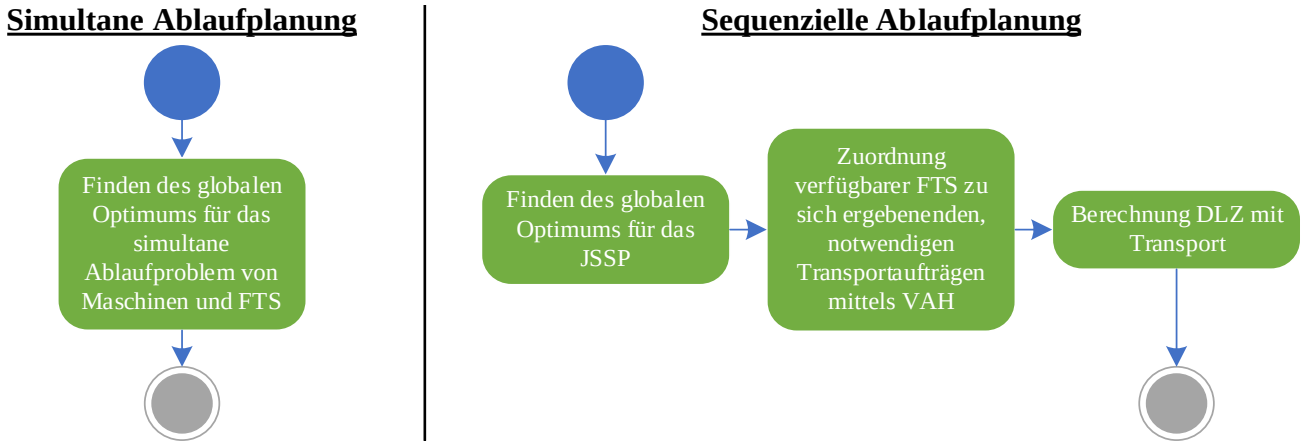


Abbildung 41: Vergleich zwischen simultaner und sequenzieller Ablaufplanung von Maschinen und FTS.

Der in [183] vorgestellte Ansatz wird für die selbstorganisierende, dezentrale Ablaufplanung zum Vergleich herangezogen. Dieser basiert auf der Nutzung eines sich rein dezentral organisierenden MAS. Eine zentrale Einheit ist entsprechend nicht vorhanden. Die genaue Funktionsweise des darin verwendeten Ansatzes ist umfangreich, weshalb für weitere Details auf die entsprechende Veröffentlichung verwiesen wird. Im Anhang M ist dennoch eine allgemeine Erläuterung des für die Interaktion und Koordination zwischen den Agenten genutzten Kontraktnetz-Verfahrens gegeben. Die Simulationsstudie in [183] wird anhand der BI von Bilge und Ulusoy durchgeführt, weshalb die Ergebnisse dieser zum hiesigen Vergleich genutzt werden können. Andere Veröffentlichungen, die eine selbstorganisierte Ablaufplanung von Maschine und FTS untersuchen und die BI von Bilge und Ulusoy nutzen, liegen nach bestem Wissen des Autors zum aktuellen Zeitpunkt nicht vor.

Die Ergebnisse des Vergleichs werden einmal für alle BI mit einem *t/p-Verhältnis* > 0,25 (Abbildung 42) und einmal für alle BI mit einem *t/p-Verhältnis* < 0,25 (Abbildung 43), betrachtet. Der Mittelwert der DLZ über alle Instanzen wird jeweils zum Vergleich herangezogen. Die Ergebnisse bzgl. der einzelnen BI sind im Anhang Q in Tabelle 24 aufgeführt. Eine DLZ-Reduzierung von durchschnittlich 19,7 %, kann durch die simultane, gegenüber der sequenziellen, Ablaufplanung, bzgl. der Instanzen mit einem *t/p-Verhältnis* > 0,25, erzielt werden. Die selbstorganisierte Ablaufplanung liefert bei diesen eine um durchschnittlich 10,4 % höhere DLZ als die sequenzielle Ablaufplanung. Die simultane Ablaufplanung resultiert gegenüber der selbstorganisierten Ablaufplanung in einer DLZ-Reduzierung von 27,2 %.

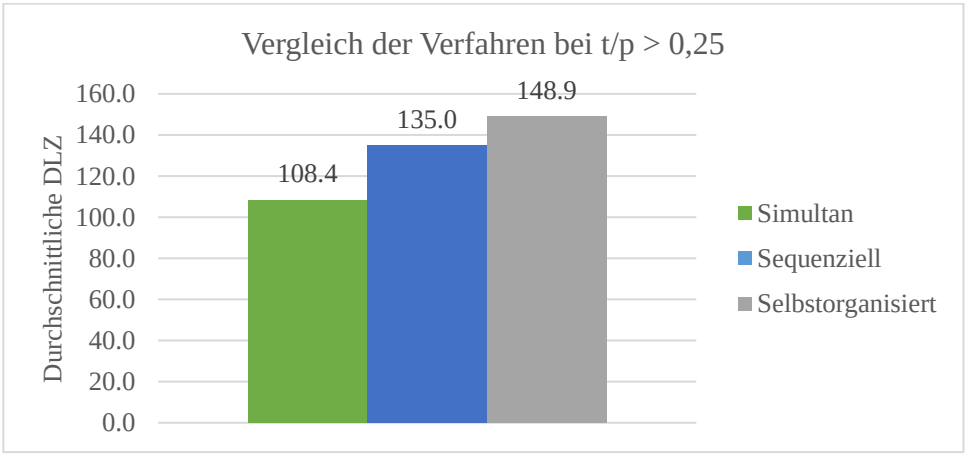


Abbildung 42: Ergebnisse des Vergleichs zwischen simultaner, sequenzieller und selbstorganisierter [183] **Ablaufplanung von Maschinen und FTS bei *t/p > 0,25*.**

Eine SAMF führt bei Instanzen mit einem *t/p-Verhältnis* < 0,25 zu einer DLZ-Reduktion von 7,0 % gegenüber der sequenziellen Ablaufplanung und von 11,4 % gegenüber der selbstorganisierten Ablaufplanung. Der selbstorganisierende Ansatz liefert ein um 5,0 % schlechteres Ergebnis als die sequenzielle Ablaufplanung. Das Optimierungspotenzial der SAMF fällt bei den Instanzen mit einem *t/p-Verhältnis* < 0,25 geringer aus als bei solchen mit einem *t/p-Verhältnis* > 0,25. Die Transportzeit besitzt bei einem *t/p* < 0,25 einen geringeren Anteil an der gesamten DLZ als bei einem *t/p* > 0,25, wodurch die Ablaufplanung der Transportaufgaben auch einen geringen Anteil an der Gesamtperformance des Systems besitzt. Das Optimierungspotenzial der SAMF ist dadurch bei einem *t/p* < 0,25 geringer.

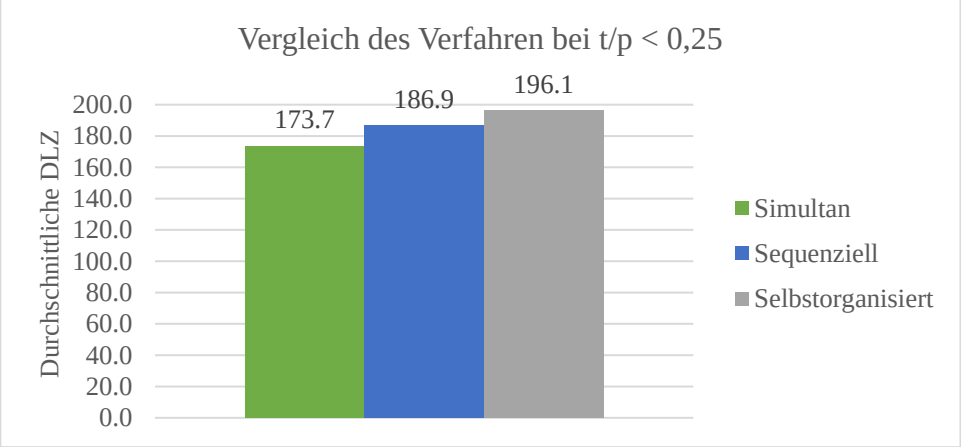


Abbildung 43: Ergebnisse des Vergleichs zwischen simultaner, sequenzieller und selbstorganisierter [183] **Ablaufplanung von Maschinen und FTS bei *t/p < 0,25*.**

Der Einsatz zentraler Ablaufplanungsverfahren (simultan und sequenziell) ermöglicht somit einer bessere DLZ als der Einsatz dezentraler Verfahren (selbstorganisierend). Ein Grund hierfür ist bei dezentralen Verfahren kein Gesamtüberblick über das System vorliegt und die Durchführung einer globalen Optimierung dadurch unwahrscheinlich ist.

Vergleich der Verfahren innerhalb einer FJSSP-Umgebung

Die drei Ablaufplanungsansätze werden folgend innerhalb einer FJSSP-Umgebung, unter der Nutzung der BI von Kumar et al. [168], verglichen. Im Rahmen der sequenziellen Ablaufplanung erfolgt zuerst die Optimierung des FJSSP, bevor anschließend den vorhandenen FTS die sich ergebenden Transportaufträge zugeordnet werden. Die Lösung des FJSSP beinhaltet, im Gegensatz zum JSSP, nicht nur die Festlegung in welcher Reihenfolge die Operationen auf den vorhandenen Maschinen bearbeitet werden, sondern auch die vorherige Selektion der Maschine, welche die jeweilige Operation bearbeiten soll. Die Zuordnung der FTS zu den einzelnen Transportaufträgen erfolgt ebenfalls durch die *VAH*. Diese wird wiederum in Java implementiert. Das CP-Modell des FJSSP ist im Anhang P zu finden. Das in Kapitel 4.1.2 vorgestellte CP-Modell zur SAMF in einer FJSSP-Umgebung sowie das CP-Modell des FJSSP werden unter Verwendung des *CP Optimizer 12.10.0* Solver gelöst. In beiden Fällen wird der Optimierungsalgorithmus solange ausgeführt, bis nachweislich das globale Optimum gefunden, oder eine RZ von 600 Sekunden überschritten ist. Die Ergebnisse bzgl. der selbstorganisierten Ablaufplanung entstammen der Veröffentlichung von Sahin et al. [168]. Der darin vorgestellte, auf MAS basierende Ansatz, stellt eine Weiterentwicklung des zum vorherigen Vergleich herangezogene Ansatz von Erol et al. [160] dar. Der Ansatz ist dahingehend weiterentwickelt, dass nun auch alternative Maschinen zur Bearbeitung einer Operation bei der Verhandlung zwischen den einzelnen Agenten berücksichtigt werden. Die in [168] veröffentlichten Simulationsergebnisse basieren ebenfalls auf den

BI von Kumar et al., wodurch die Vergleichbarkeit zwischen den drei Verfahren gegeben ist. Andere Veröffentlichungen, welche die selbstorganisierte Ablaufplanung von Maschine und FTS untersuchen und die BI von Kumar et al. nutzen, liegen nach bestem Wissen des Autors zum aktuellen Zeitpunkt nicht vor.

Die erzielten Simulationsergebnisse sind in Abbildung 44 aufgeführt. Bei den aufgeführten Werten handelt es sich jeweils um die durchschnittliche DLZ über alle 40 BI. Eine Aufführung der detaillierten Ergebnisse für jede der 40 BI und jedes der drei Verfahren ist im Anhang Q in Tabelle 25 zu finden. Die SAMF ermöglicht eine DLZ-Reduzierung von 35,6 % gegenüber dem sequenziellen und von 40,3 % gegenüber dem selbstorganisierenden Ansatz. Das Optimierungspotenzial der simultanen gegenüber der sequenziellen Ablaufplanung fällt bei dem Vorhandensein alternativer Maschine mit 35,6 % höher aus, als wenn keine alternativen Maschinen vorhanden sind (19,7 %).

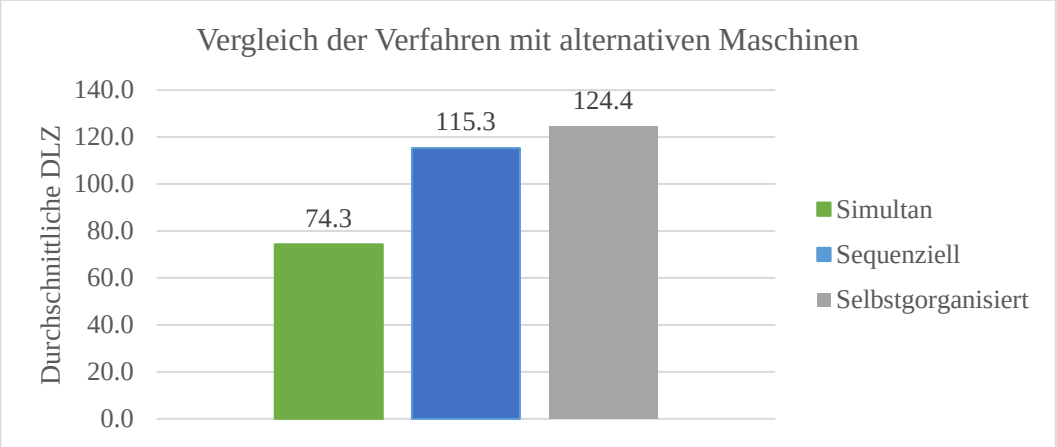


Abbildung 44: Ergebnisse des Vergleichs zwischen simultaner, sequenzieller und selbstorganisierter [168] **Ablaufplanung von Maschinen und FTS bei *t/p > 0,25* und alternativen Maschinen (FJSSP).**

Die 1. Forschungsfrage *„Bewirkt die SAMF eine DLZ-Reduktion gegenüber der sequenziellen sowie der selbstorganisierenden Ablaufplanung von Maschinen und FTS?“* kann durch die Ergebnisse dieser Simulationsstudie positiv beantwortet werden.

4.3.3 Vergleich der Nutzung von CP zur SAMF gegenüber anderen State-of-the-Art Verfahren

In diesem Kapitel erfolgt die Gegenüberstellung des CP-basierten Optimierungsansatzes mit anderen, ihm Rahmen unterschiedlicher Arbeiten entwickelten, Optimierungsalgorithmen für die SAMF. Die Simulationsstudien in den einzelnen, zum Vergleich herangezogenen Arbeiten, wurden ebenfalls auf Grundlage der BI von Bilge und Ulusoy [158] für JSSP-Umgebungen, respektive von Kumar et al. [167] für FJSSP-Umgebungen, durchgeführt. In den meisten Arbeiten wird entweder eine JSSP- oder eine FJSSP-Umgebung betrachtet. Die einzelnen folgend aufgelisteten Ergebnisse sind den jeweiligen Publikationen entnommen. Als Vergleichskriterien dienen die DLZ und die RZ. Die RZ ist allerdings nicht in allen, zum Vergleich herangezogenen Arbeiten, angegeben. Besonders in RS, in denen häufig unerwartete Ereignisse wie Maschinenausfälle, sonstige Störungen, oder Planungsänderungen auftreten, ist es essenziell, dass der verwendete Optimierungsalgorithmus zur Ablaufplanung in angemessener Zeit eine gute, neue und gültige Lösung liefern kann.

Das jeweilige CP-Modell ist in Java implementiert und wird unter Verwendung des Solvers *CP Optimizer* gelöst. Als Hardware wird eine *Intel® Core™ i5-7200U* CPU mit 2,50 GHz sowie 8,00 GB

RAM verwendet. Für jede einzelne BI werden zehn Optimierungsdurchläufe ausgeführt und der durchschnittliche Wert als Ergebnis zum Vergleich herangezogen.

Es folgt zuerst ein Vergleich des entwickelten CP-basierten Ansatzes, mit den Arbeiten, die eine JSSP-Umgebung zugrunde legen. Anschließend wird der Ansatz mit denen die eine FJSSP-Umgebung zugrunde legen verglichen.

Vergleich mit State-of-the-Art Verfahren innerhalb einer JSSP-Umgebung

Der Vergleich zwischen dem in dieser Arbeit vorgestellten CP-Verfahren zur SAMF mit anderen publizierten Arbeiten zur Optimierung dieser, erfolgt in diesem Abschnitt. In Tabelle 10 sind die Ergebnisse für die Instanzen aus [158] mit einem Verhältnis von $t/p > 0{,}25$ und in Tabelle 11 die Ergebnisse für Instanzen mit einem $t/p < 0{,}25$ aufgeführt. Die Durchschnittswerte über alle BI werden darin jeweils aufgeführt. Die ausführlichen Ergebnisse sind im Anhang Q in Tabelle 26 ($t/p > 0{,}25$) und Tabelle 27 ($t/p < 0{,}25$) zu finden.

Die zum Vergleich herangezogenen Verfahren werden im Folgenden nochmals kurz erläutert. Erol et al. [160] nutzen einen auf MAS-basierenden Ansatz, wobei eine reine Selbstorganisation durchgeführt wird. Lin et al. [170] verwenden einen Ansatz der simulationsbasierten Optimierung (SBO), betrachten allerdings lediglich die Instanzen mit einem $t/p > 0{,}25$. Zheng et al. [184] sowie Chaudhry und Shami [162] nutzen mit TS respektive GA jeweils (Meta-)heuristische Verfahren zur Optimierung der SAMF. Huang [185] verwendet, als Vertreter exakter Verfahren, MILP zur Lösung der simultanen Ablaufplanung. In [160], [170] und [184] wird kein RZ veröffentlicht. In [162] und [185] wird die RZ lediglich für die Instanzen mit $t/p > 0{,}25$ veröffentlicht.

	Erol [160]	Lin [170]	Zheng [196]	Chaudhry [162]		Huang [185]		Groß	
	MAS	SBO	TS	GA		MILP		CP	
	DLZ	DLZ	DLZ	DLZ	RZ [sec.]	DLZ	RZ [sec.]	DLZ	RZ [sec.]
Ø	**148,9**	**108,8**	**108,4**	**109,4**	**122,2**	**107,8**	**6162,9**	**108,4**	**11,5**
Median					**36**		**116,205**		**0,36**

Tabelle 10: Vergleich der Ergebnisse verschiedener Optimierungsverfahren für die SAMF ($t/p > 0{,}25$).

Bzgl. der Ergebnisse von [185] ist zu beachten, dass dieses Verfahren nicht für alle Instanzen eine Lösung findet. Durch diesen Umstand ergibt sich eine bessere durchschnittliche DLZ gegenüber dem in dieser Arbeit vorgestellten Verfahren. Die Ergebnisse für die einzelnen, gelösten Instanzen sind bei [185] und dem vorgestellten Ansatz jedoch identisch. Die von Erol [160] präsentierten Ergebnisse liegen deutlich hinter den Ergebnissen der anderen Ansätze zurück. Die Ansätze von Lin [176], Zheng [196] und Chaudhry und Shami [162] liefern jeweils ähnliche Ergebnisse. Die RZ von [162] ist mit einem durchschnittlichen Wert von 122,2 s sowie einem Median von 36 s deutlich besser als die Werten aus [185] mit einer durchschnittlichen RZ von 6162,9 s, respektive einem Median von 116,2 s. Der im Rahmen dieser Arbeit vorgestellte Ansatz liefert sowohl bzgl. der DLZ als auch der RZ die besten Ergebnisse. Er liefert mit einer durchschnittlichen DLZ von 108,4 die gleichen Ergebnisse wie Zheng [196]. Bezogen auf die RZ ist er [162] und [185], mit einer durchschnittlichen RZ von 11,5 s sowie einem Median von 0,36 s, deutlich überlegen.

Der CP-Ansatz erreicht bei den Instanzen mit $t/p < 0{,}25$ eine durchschnittliche DLZ von 173,7 sowie eine durchschnittliche RZ von 1,3 s mit einem Median von 0,03 s. Der GA-Ansatz von Chaudhry [162]

erreicht ebenfalls eine DLZ von 173,7, es wird allerdings keine RZ veröffentlicht. Erol et al. [160] liefern mit 196,1 die schlechteste DLZ dieses Vergleichs.

	Erol [160]	Chaudhry [162]	Groß	
	MAS	GA	CP	
	DLZ	DLZ	DLZ	RZ [sec.]
Ø	**196,1**	**173,7**	**173,7**	**1,3**
Median				**0,03**

Tabelle 11: Vergleich der Ergebnisse verschiedener Optimierungsverfahren für die SAMF *(t/p < 0,25).*

Vergleich mit State-of-the-Art Verfahren innerhalb einer FJSSP-Umgebung

Im Folgenden wird der vorgestellte CP-Ansatz zur SAMF in einer FJSPP-Umgebung anderen in der Literatur veröffentlichen Optimierungsalgorithmen gegenübergestellt. Simulationsergebnisse bzgl. der SAMF mit alternativen Maschinen und unter der Verwendung der von Kumar et al. [167] vorgestellten BI, werden nach bestem Wissen des Autors, nur in den Veröffentlichungen von Kumar et al. [167] selbst, von Lin et al. [170] und von Sahin et al. [168] präsentiert. Die jeweils in den einzelnen Veröffentlichungen publizierten Simulationsergebnisse sowie die durch den vorgestellten Ansatz erzielten Simulationsergebnisse, sind in Tabelle 12 aufgelistet. Bei den Werten handelt es sich um die Durchschnittswerte aller 40 BI. Der vorgestellte Optimierungsansatz liefert mit einer DLZ von 74,3 das beste Ergebnis. Die RZ beträgt hierbei durchschnittlich 9,35 Sekunden und im Median 4,24 Sekunden. In den zum Vergleich herangezogenen Veröffentlichungen werden keine RZ veröffentlicht. Die ausführlichen Ergebnisse sind im Anhang Q in Tabelle 28 aufgeführt.

	Kumar [167]		Lin [170]	Sahin [168]	Groß	
	PDE-1	PDE-2	$L\text{-}GA_{OCBA}$	FMAS	CP	
	DLZ	DLZ	DLZ	DLZ	DLZ	RZ [sec.]
Ø	**77,5**	**77,1**	**115,0**	**124,4**	**74,3**	**3,9**
Median						**1,26**

Tabelle 12: Vergleich der Ergebnisse verschiedener Optimierungsverfahren für die SAMF mit alternativen Maschinen.

Die Ansätze von Lin et al. und Sahin et al., die auf simulationsbasierter Optimierung respektive reiner Selbstorganisation durch ein MAS basieren, liefern eine deutlich schlechtere DLZ, als der zentrale Ablaufplanungsansatz von Kumar und der in dieser Arbeit vorgestellte Ansatz. Im Vergleich zu Kumar (PDE-2) ergibt sich bei Lin et al. eine 49,1 % höhere DLZ und bei Sahin et al. eine um 61,3 % höhere DLZ. Lin et al. sowie Sahin et al. sind allerdings die einzigen beiden, zum Vergleich herangezogenen, Ansätze, die auch auf unerwartet auftretende Ereignisse reagieren können. Dies zeigt, dass hier die Fähigkeit zur Reaktion und Anpassung auf solche Ereignisse zulasten der Lösungsqualität geht.

Der in dieser Arbeit vorgestellte Ansatz kann hingegen sowohl auf unerwartet eintretende Ergebnisse reagieren, als auch eine sehr gute Lösungsqualität liefert. Die Reaktionsfähigkeit des vorgestellten Ansatzes auf unerwartet eintretende Ereignisse wird in Kapitel 4.1.3 untersucht und verifiziert.

Die 2. Forschungsfrage *„Kann die SAMF, trotz der hohen Komplexität des Problems, in einer akzeptablen Rechenzeit erfolgen?“* kann durch die Ergebnisse dieser Simulationsstudie positiv beantwortet werden. Der CP-basierte Ansatz zur SAMF ermöglicht die geringste RZ gegenüber den zum Vergleich herangezogenen Verfahren. Diese liegt bei den untersuchten BI innerhalb einer JSSP-Umgebung im

Median bei deutlich unter einer Sekunde und innerhalb einer FJSSP-Umgebung im Median unter 5 sec. Sollte die RZ für den jeweiligen Einsatzzweck, dennoch zu hoch sein, kann die Optimierung auch zu einem vorherigen Zeitpunkt abgebrochen und der bis dahin beste, gefundene Ablaufplan genutzt werden.

4.3.4 Abhängigkeit zwischen DLZ-Reduktion und t/p-Verhältnis

Die vorherigen Untersuchungen zeigen, dass das Optimierungspotenzial durch die SAMF bei den BI mit einem *t/p*-Verhältnis > 0,25 höher ausfällt, als bei den Instanzen mit einem *t/p* < 0,25. Aufgrund dessen wird in diesem Kapitel der Zusammenhang zwischen *t/p*-Verhältnis und dem Optimierungspotenzial zur DLZ-Reduktion durch SAMF untersucht. Die prozentuale, durch SAMF gegenüber sequenzieller Ablaufplanung erzielte, DLZ-Reduktion ist in Abbildung 45 (*t/p* < 0,25) und Abbildung 46 (*t/p* > 0,25) in Abhängigkeit des jeweiligen *t/p*-Verhältnisses dargestellt. Die Ergebnisse der Simulationsstudie in Kapitel 4.2.1 werden hierzu genutzt. Jeder blaue Punkt stellt eine der 82 BI mit der zugehörigen DLZ-Reduktion (Ordinate) und dem zugehörigen *t/p*-Verhältnis (Abszisse) dar. Die jeweilige, mittels einfacher, linearer Regression berechnete, Trendlinie, ist als gestrichelte Linie dargestellt. Diese zeigt in beiden Abbildungen, dass ein steigendes *t/p*-Verhältnis mit einer Erhöhung der prozentualen DLZ-Reduktion einhergeht.

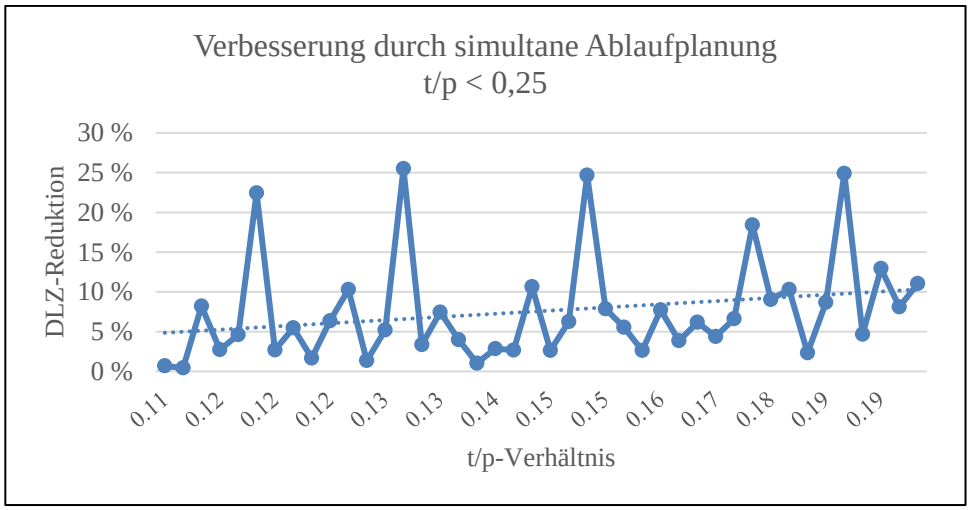


Abbildung 45: Zusammenhang zwischen DLZ-Reduktion durch SAMF und dem t/p Verhältnis für alle Instanzen von Bilge und Ulusoy mit t/p < 0,25.

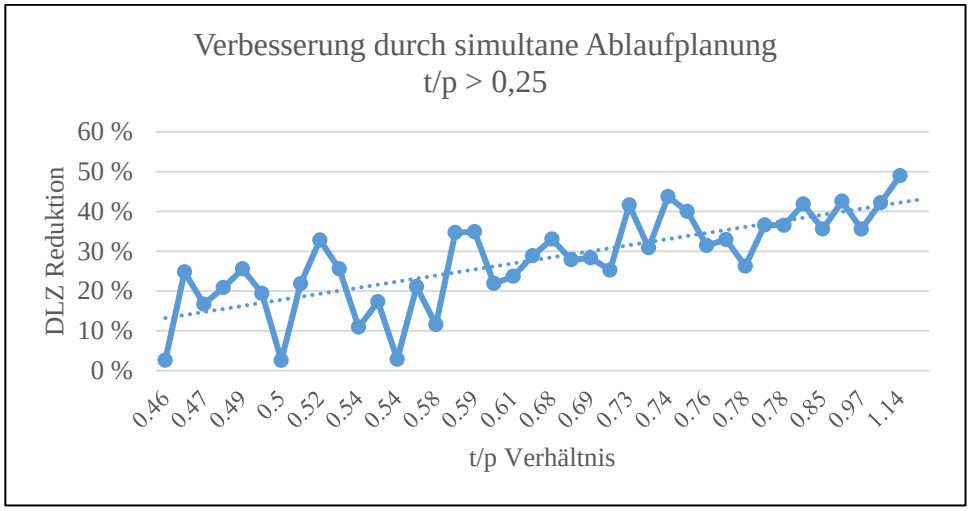


Abbildung 46: Zusammenhang zwischen DLZ-Reduktion durch SAMF und dem t/p Verhältnis für alle Instanzen von Bilge und Ulusoy mit t/p > 0,25

Die DLZ setzt sich bei realen Werkstattfertigungen (abbildbar durch eine JSSP-Umgebung) folgendermaßen zusammen [186]:

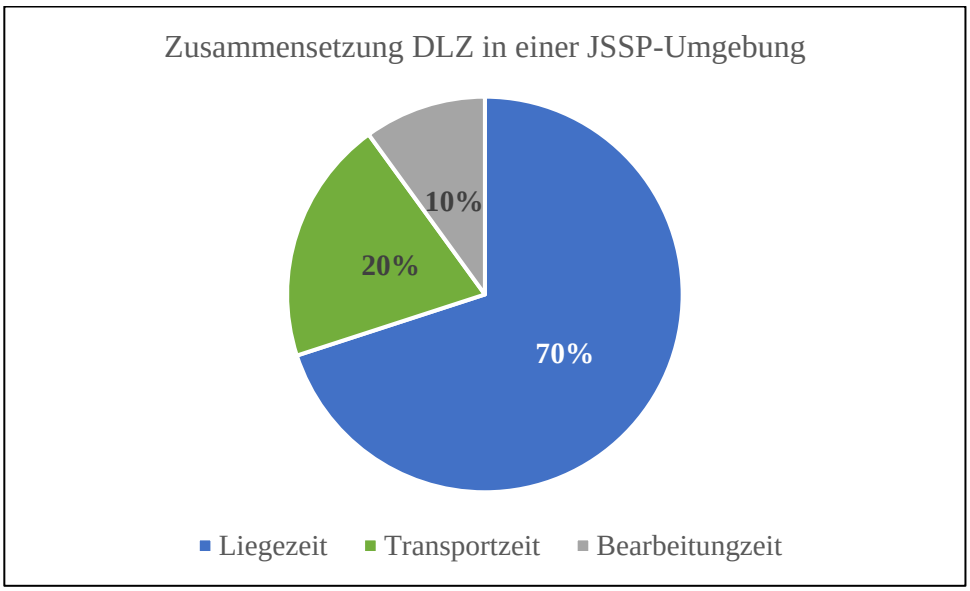


Abbildung 47: Zusammensetzung der DLZ in einer realen DLZ-Umgebung nach [186].

Dies entspricht einem *t/p*-Verhältnis von 2,0, was deutlich über den Werten der betrachteten BI liegt. Durch den aufgezeigten Zusammenhang zwischen steigendem *t/p*-Verhältnis und steigender DLZ-Reduktion, kann die DLZ-Reduktion durch SAMF in realen RS höher ausfallen, als bei den hier untersuchten BI.

4.3.5 Abhängigkeit zwischen DLZ und Anzahl verfügbarer FTS

Eine wichtige Frage beim Einsatz von FTS in der Intralogistik, ist die Anzahl an notwendigen FTS, um einen möglichst effizienten Materialfluss gewährleisten zu können. In diesem Kapitel erfolgt hierzu die Untersuchung des Einflusses verschiedener Faktoren auf die Anzahl notwendiger FTS. Folgende, mögliche Abhängigkeiten werden untersucht:

- Abhängigkeit zwischen Anzahl an FTS und DLZ

- Abhängigkeit notwendiger FTS und *t/p*-Verhältnis
- Abhängigkeit notwendiger FTS und Anzahl an Aufträgen innerhalb des Fertigungssystems

Die Abhängigkeit zwischen DLZ und der Anzahl an FTS ist in Abbildung 48 dargestellt. Für diese Untersuchung werden die BI von Bilge und Ulusoy [158] verwendet. Alle darin enthaltenen Instanzen mit einem *t/p*-Verhältnis > 0,25 werden mit einer variierenden Anzahl verfügbarer FTS optimiert. Der in Kapitel 4.1.1 vorgestellte Algorithmus zur SAMF wird für diese Optimierung genutzt und die Anzahl verfügbarer FTS variiert zwischen einem und acht FTS. Der Optimierungsalgorithmus wird solange ausgeführt bis das globale Optimum für die vorliegende Instanz mit der jeweiligen verfügbaren Anzahl an FTS gefunden ist. Die in Abbildung 48 jeweils zu sehende DLZ (blaue Punkte) entspricht der durchschnittlichen DLZ über alle BI, wenn die jeweils auf der Abszisse aufgeführten Anzahl an FTS verfügbar ist.

Die DLZ verbessert sich mit steigender Anzahl an FTS zuerst stark. Ab einer gewissen Anzahl verfügbarer FTS, im vorliegenden Fall ab fünf FTS, ergibt sich allerdings keine weitere Reduzierung der DLZ mehr. Die minimale DLZ wird erreicht, wenn keine der vorhandenen Maschinen eine Leerlaufzeit hat, die sich auf das Warten auf ein zu bearbeitendes Produkt zurückführen lässt. Somit muss sichergestellt werden, dass genügend FTS vorhanden sind, um das Eintreten dieses Falls zu verhindern. Eine weitere Erhöhung der Anzahl verfügbarer FTS bringt ab diesem Punkt keine weitere Verbesserung bzgl. der DLZ-Reduktion.

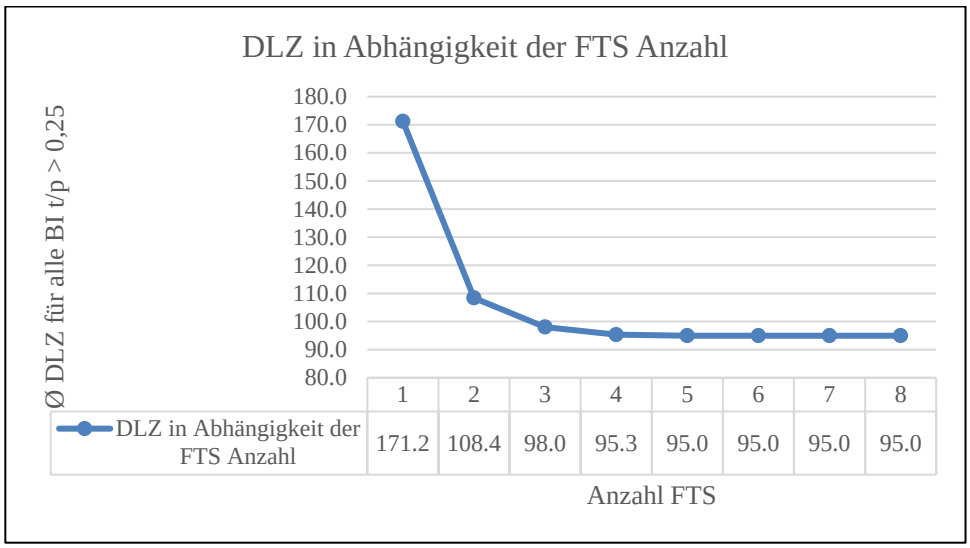


Abbildung 48: Durchschnittliche DLZ in Abhängigkeit der Anzahl an verfügbaren FTS mit den BI von Bilge und Ulusuoy.

In Abbildung 49 ist die mindestens notwendige Anzahl an FTS, zur Erreichung einer minimalen DLZ, in Abhängigkeit des *t/p*-Verhältnisses der jeweiligen BI aufgetragen. Jeder Punkt entspricht einer der 40 BI aus [158] mit einem *t/p*-Verhältnis > 0,25. Die Ergebnisse der vorherigen Simulationsstudie dienen als Grundlage für diese Untersuchung. Bei der gestrichelten Linie handelt es sich um eine Trendlinie. Diese zeigt auf, dass je höher das *t/p*-Verhältnis ist, desto mehr FTS werden benötigt, um eine minimale DLZ zu erreichen. Eine Begründung hier ist, dass mit steigendem *t/p*-Verhältnis der Anteil der Bearbeitungszeit an der Gesamtzeit abnimmt und der Anteil der Transportzeit hingegen zunimmt. Hierdurch kommt es bei gleicher Anzahl an vorhandenen FTS irgendwann dazu, dass eine Maschine mit dem Beginn der Bearbeitung einer Operation warten muss, bis das Produkt zur entsprechenden Maschine transportiert wurde. Dadurch muss mit steigendem *t/p*-Verhältnis auch die Anzahl an verfügbaren FTS erhöht werden, um eine minimale DLZ realisieren zu können.

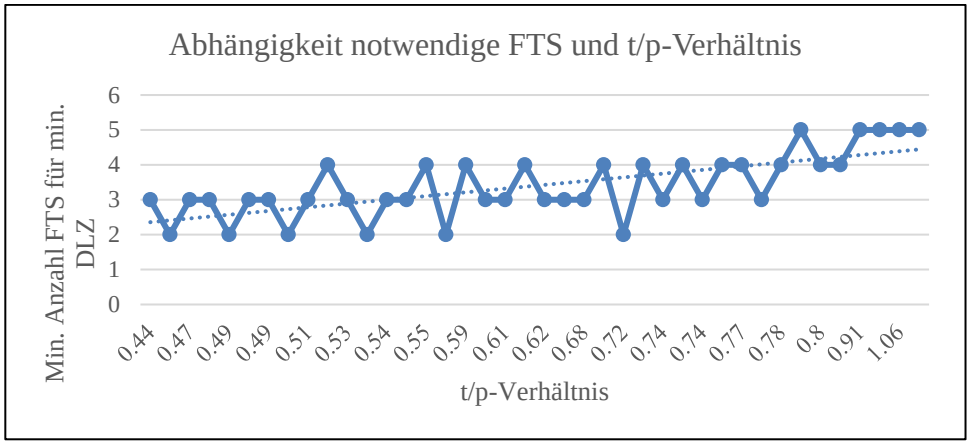


Abbildung 49: Abhängigkeit zwischen der Anzahl an FTS, ab welchen keine Reduzierung der DLZ mehr erreicht wird und dem *t/p*-Verhältnis der jeweiligen BI.

Die Gegenüberstellung der mindestens notwendigen Anzahl an FTS, zur Erreichung einer minimalen DLZ und der Anzahl an Aufträgen innerhalb des Fertigungssystems ist in Abbildung 50 aufgetragen. Jeder einzelne Punkt entspricht auch hier einer der 40 BI aus [158] mit einem *t/p*-Verhältnis > 0,25. Die gestrichelte Linie entspricht der Trendlinie. Mit einer Zunahme an Aufträgen innerhalb des RS, ist weder ein deutlicher Anstieg noch Abfall bzgl. der Anzahl an FTS, die mindestens notwendig ist, um eine minimale DLZ zu erreichen, zu verzeichnen.

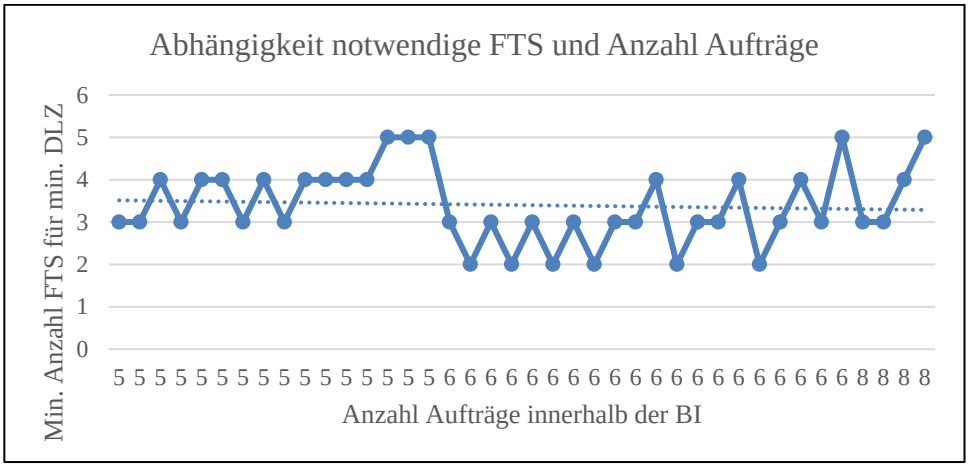


Abbildung 50: Abhängigkeit zwischen der Anzahl an FTS, ab welchen keine Reduzierung der DLZ mehr erreicht wird und der Anzahl an Aufträge innerhalb der jeweiligen BI.

Das *t/p*-Verhältnis ist in Abhängigkeit der Anzahl an Aufträgen innerhalb der jeweiligen BI in Abbildung 51 aufgetragen. Jeder einzelne Punkt entspricht auch hier einer der 40 BI aus [158] mit einem *t/p*-Verhältnis > 0,25 und die gestrichelte Linie entspricht der Trendlinie. Eine eindeutige Abhängigkeit zwischen dem *t/p*-Verhältnis und der Anzahl an Aufträgen innerhalb einer BI ist nicht zu erkennen.

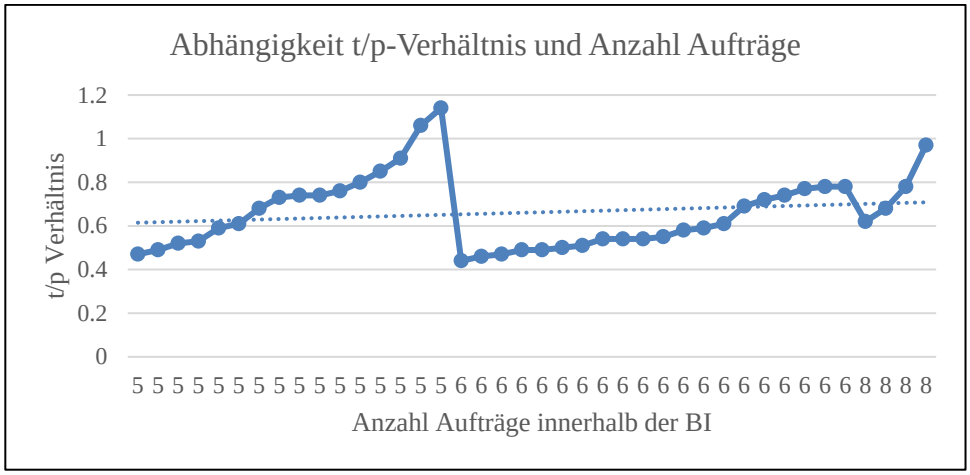


Abbildung 51: Abhängigkeit zwischen dem *t/p*-Verhältnis und der Anzahl an Aufträge innerhalb eines der 40 Auftragssätze bzgl. der BI von [158].

Eine Abhängigkeit zwischen der mindestens notwendigen Anzahl an FTS, zur Erreichung einer minimalen DLZ und dem *t/p*-Verhältnis ist vorhanden. Die Anzahl notwendiger FTS steigt mit dem *t/p*-Verhältnis an. Eine Abhängigkeit zwischen der mindestens notwendigen Anzahl an FTS, zur Erreichung einer minimalen DLZ und der Anzahl an zu bearbeitenden Aufträgen (Produkte), kann nicht nachgewiesen werden. Des Weiteren ist festzustellen, dass eine Erhöhung der Anzahl an verfügbaren FTS ab einem gewissen Punkt nicht mehr zu einer weiteren Reduzierung der DLZ führt. Berücksichtigt man zudem die Problematik von Routenkonflikten, kann eine weitere Erhöhung der Anzahl an vorhanden FTS in einem realen Szenario ab diesem Punkt auch potenziell zu einer Erhöhung des DLZ führen. Ein Routenkonflikt kann bspw. auftreten, wenn zwei oder mehr FTS zur gleichen Zeit an einer Engstelle aufeinandertreffen und keines der betroffen FTS seine Fahrt fortsetzen kann. Dieses Event wird als Deadlock bezeichnet. Ein Werker muss hierbei das Problem meist durch manuelle Steuerungseingriff am FTS lösen.

4.3.6 Vergleich unterschiedlicher Zielfunktionen

Die DLZ wird in vielen Benchmarks für Ablaufplanungsprobleme als Zielfunktion genutzt. Auch in den zuvor durchgeführten Simulationsstudien wurde diese als Optimierungskriterium genutzt. Im Rahmen der folgenden Simulationsstudie werden vier verschiedene Zielfunktionen sowie deren Auswirkungen auf unterschiedliche KPIs innerhalb der SAMF verglichen:

- DLZ
- MA
- Multikriteriell, lexikografisch: Kriterium 1: DLZ, Kriterium 2: MA
- Multikriteriell, lexikografisch: Kriterium 1: MA, Kriterium 2: DLZ

Die KPIs DLZ, MA sowie die Auslastung jeder einzelnen vorhandenen Maschine (*M1, …, M4*) werden in Abhängigkeit der genutzten Zielfunktion betrachtet. Als Simulationsumgebung dienen die BI von Kumar et al. und als verwendeter Optimierungsalgorithmus das in Kapitel 4.1.2 vorgestellte CP-Modell zur SAMF in einer FJSSP-Umgebung. Alle 40 BI werden bzgl. jeder Zielfunktion optimiert. Die Optimierung wird nach 120 Sekunden RZ abgebrochen und die bis dahin beste gefundene Lösung gewertet. Die Ergebnisse sind als Durchschnittswerte über alle 40 BI in Tabelle 13 aufgeführt. Ein Einfluss der unterschiedlichen Zielfunktionen auf die KPIs ist nur in geringem Ausmaß festzustellen. Die beste DLZ liegt 4,4 % unter der schlechtesten DLZ. Die Differenz zwischen der geringsten und

höchsten MA beträgt 0.93 Prozentpunkte. Ein Grund hierfür ist, dass zwischen der MA und DLZ eine Abhängigkeit besteht. Für die Berechnung der MA wird die DLZ als Gesamtzeit (Soll-Leistung) betrachtet. Die MA ergibt sich als Ergebnis der Division der tatsächlichen Produktionszeit (Ist-Leistung) der jeweiligen Maschine durch die Soll-Leistung. Eine Reduzierung der DLZ, unter Annahme einer gleichbleibenden tatsächlichen Produktionszeit, führt somit zu einer Erhöhung der MA. Hierdurch kann auch begründet werden, weshalb die MA bei Verwendung der multikriteriellen, lexikografischen Zielfunktion (Kriterium 1: DLZ; Kriterium 2: MA) eine höhere MA erzielt, als bei alleiniger Verwendung der MA als Zielfunktion.

	KPIs					
Verwendete Zielfunktion	DLZ	MA	MA *M1*	MA *M2*	MA *M3*	MA *M4*
DLZ	**74.30**	**67.27 %**	**68.47 %**	**73.20 %**	**70.11 %**	**57.29 %**
MA	**77.70**	**67.76 %**	**71.34 %**	**71.85 %**	**70.65 %**	**57.20 %**
Multikriteriell (DLZ, MA)	**75.15**	**68.20 %**	**73.20 %**	**70.95 %**	**69.51 %**	**59.14 %**
Multikriteriell (MA, DLZ)	**77.58**	**67.63 %**	**72.40 %**	**70.02 %**	**70.03 %**	**58.09 %**

Tabelle 13: Vergleich unterschiedlicher ein- und multikriterieller Zielfunktionen für die simultane Ablaufplanung basierend auf den BI von Kumar et al.

In den zuvor aufgezeigten Simulationsstudien wurde der CP-basierte Ansatz zur SAMF-Optimierung mit der sequenziellen und selbstorganisierten Ablaufplanung verglichen. Die SAMF konnte hierbei die besten Ergebnisse bzgl. der DLZ-Minimierung erzielen. Der Ansatz wurde des Weiteren anderen Optimierungsverfahren zur SAMF gegenübergestellt. Hierbei konnte gezeigt werden, dass der CP-basierte Ansatz die besten Ergebnisse bzgl. DLZ und RZ liefert. Zudem wurde die Abhängigkeit zwischen dem Potenzial der DLZ-Reduktion durch SAMF in Abhängigkeit des vorliegenden *t/p*-Verhältnisse untersucht. Es wurde gezeigt, dass das Potenzial der DLZ-Reduktion durch SAMF mit steigendem *t/p*-Verhältnisse ebenfalls zunimmt. Abschließend erfolgte die Untersuchung der Abhängigkeit zwischen DLZ und Anzahl verfügbarer FTS. In jeder dieser Simulationsstudien wurde eine deterministische Umgebung angenommen. Im Folgenden wird deshalb die Reaktionsfähigkeit des Ansatzes auf unerwartet auftretende Ereignisse in einer nicht deterministischen Umgebung untersucht.

4.4 Reaktion der hybriden Steuerungsarchitektur auf unerwartet auftretende Ereignisse

In RS treten häufig unerwartet Ereignisse auf. Das Steuerungssystem muss hier in der Lage sein, schnell und adäquat auf die geänderte Situation reagieren zu können und dieser mit der Erstellung eines neuen, gültigen Ablaufplans sowie entsprechenden Steuerungsaktionen entgegenzuwirken. Ein Echtzeit-Abbild der Fertigungsebene muss vorhanden sein, um die genaue und aktuelle Situation zu kennen. Zumdem muss der verwendete Ablaufplanungsalgorithmus in der Lage sein, innerhalb kurzer Zeit einen neuen, optimierten und gültigen Ablaufplan zu erzeugen. Mit Ausnahme von zwei der vorgestellten Veröffentlichungen zur SAMF, betrachtet allerdings keine dieser die Reaktion des jeweiligen Ansatzes auf unerwartet eintretende Ereignisse. Es wird lediglich von einer idealisierten, deterministischen Umgebung ausgegangen. Die Entwicklung der Algorithmen für solche deterministischen Fälle reicht zur Steuerung eines RS allerdings nicht aus. Die Reaktionsfähigkeit des vorgestellten Ansatzes soll aus diesem Grund folgend in einer nicht deterministischen Umgebung, mit unerwartet auftretenden Ereignissen, untersucht werden. Die Reaktion und Neuplanung erfolgt nicht alleinig durch das vorgestellte CP-Modell innerhalb der Scheduler Komponente, sondern durch die Kommunikation

und Interaktion aller Komponenten der hybriden Steuerungsarchitektur. Als Zielfunktion für die folgenden Untersuchungen dient die DLZ. Die Optimierung wird nach einer Sekunde abgebrochen und die bis dahin beste gefundene Lösung als Ergebnis verwendet, da beim Auftreten unvorhergesehener Ereignisse eine schnelle Reaktion notwendig ist.

Innerhalb dieser Simulationsstudie werden folgende, während der Prozessausführung unerwartet auftretende, Ereignisse untersucht:

- Ausfall einer Maschine
- Ausfall eines FTS
- Zu berücksichtigende Inspektionsergebnisse (neue, zusätzlich durchzuführende Operation)
- Neue, auszuführende Aufträge (zu (re-)fabrizierende Produkte)

Diese Ereignisse sind dem Steuerungssystem vorher nicht bekannt, sodass es sich aus dessen Sicht um unerwartet auftretende Ereignisse handelt. Der Auftragssatz 1 sowie des Layout 1 der BI von Kumar et al. werden als Simulationsszenario genutzt. Die Auswahl dieser BI ist beliebig. Der von der Steuerungsarchitektur für diese BI erzeugte, initiale Ablaufplan ist in Abbildung 52 dargestellt. Während der Ausführung dieses initialen Ablaufplans werden die unerwartet auftretenden Ereignisse simuliert. Das Eintreten eines solchen Ereignisses löst eine Neuplanung aus. Der initiale Ablaufplan sowie die während dieser Simulationsstudien folgenden Ablaufpläne sind analog zu dem Ablaufplan in Abbildung 22 zu verstehen. Alle Ereignisse bzgl. des Beginns und des Endes der Transportfahrten und Bearbeitungsprozesse werden der Steuerungsarchitektur während der Simulationsdurchführung als simulierte Signale übermittelt. Diese ist somit in der Lage ihre Zustandsmodelle an den aktuellen Stand der Prozessausführung anzupassen. Die in Abbildung 52 zu sehenden, geplanten Start- und Endzeitpunkte der Bearbeitungen der Operationen sowie der Transportfahrten werden zur Übermittlung der entsprechenden Signale verwendet. Im realen Einsatz erfolgt die Übermittlung dieser Signale über die jeweiligen Transport -, Expert - und Ressourcen Komponenten, wenn eine Transportfahrt bzw. die Bearbeitung einer Operation startet, oder endet. Ein realitätsnaher Einsatz der Steuerungsarchitektur kann durch dieses Vorgehen simuliert werden. Zudem kann die Reaktion dieser, auf im laufenden Prozess, unerwartet auftretender Ereignisse, untersucht werden. Die Ereignisse können einzeln, immer bezogen auf den initialen Ablaufplan (Abbildung 52), aber auch als Folge mehrerer unerwarteter Ereignisse nacheinander eintreten.

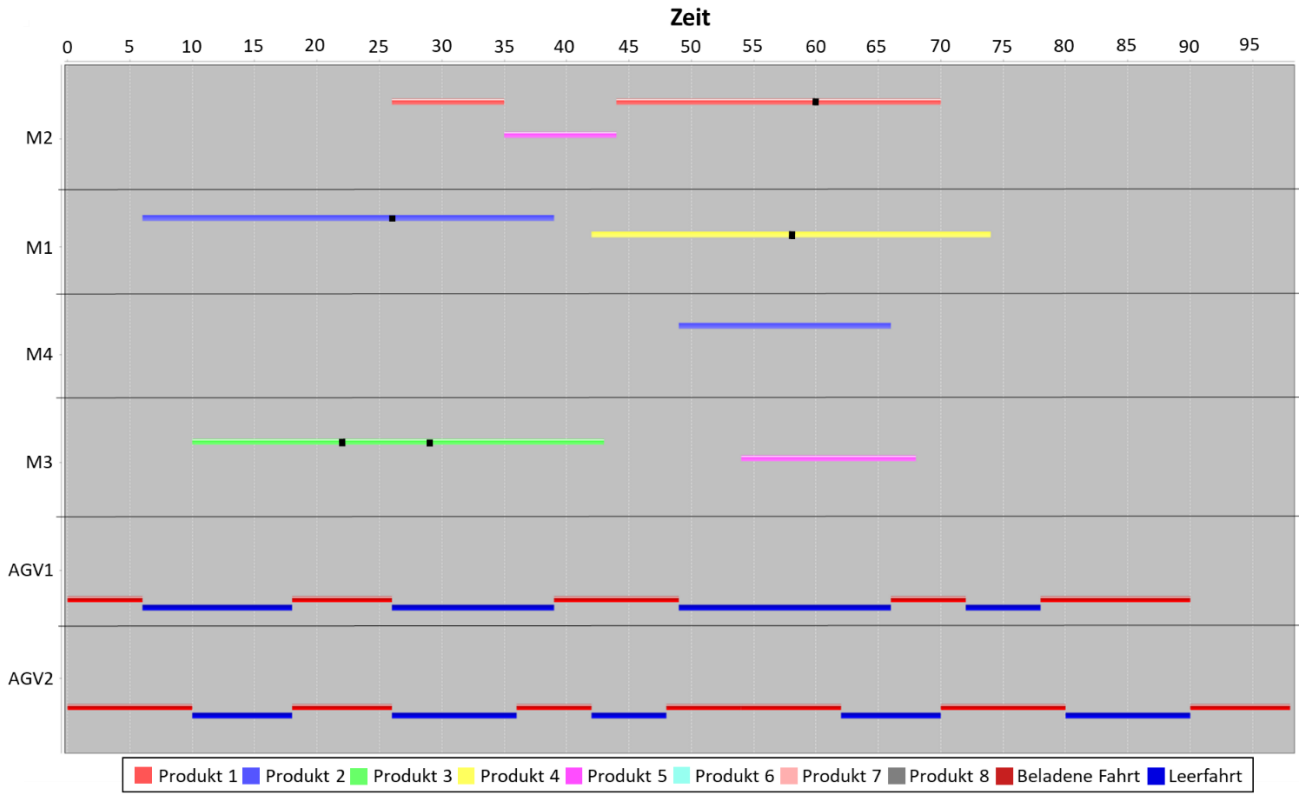


Abbildung 52: Simultaner Ablaufplan für Auftragssatz 1 und Layout 1 der BI von Kumar et al.

4.4.1 Reaktion auf Ausfall einer Maschine und auf Ausfall eines FTS

In diesem Abschnitt wie die Reaktion der Steuerungsarchitektur auf den Ausfall einer Maschine sowie eines FTS untersucht. Sowohl der Maschinenausfall, als auch der Ausfall eines der beiden FTS, wird 15 Zeiteinheiten (ZE) nach dem Beginn der Ausführung des Ablaufplans in Abbildung 52 simuliert. Die beiden Ereignisse werden getrennt voneinander simuliert und betrachtet. Die, im Rahmen einer entsprechenden Neuplanung erzeugten, Ablaufpläne werden zur Untersuchung betrachtet.

Der als Reaktion auf den Ausfall von Maschine *M2* erzeugte Ablaufplan ist in Abbildung 53 zu sehen. Die jeweils erste Operation von Produkt 2 und 3 ist im neuerzeugten Ablaufplan nicht mehr enthalten. Diese befinden sich zum Zeitpunkt des Maschinenausfalls bereits in Bearbeitung und werden somit nicht neugeplant. Die sich ergebenden Verfügbarkeiten der Produkte 2 und 3 sowie die der bearbeitenden Maschinen werden entsprechend berücksichtigt. Zudem werden alle zuvor auf Maschine *M2* geplanten Operationen auf andere Maschinen umgeplant. Alle Operationen von Produkt 1 werden auf *M3* und *M4* aufgeteilt. Die erste Operation von Produkt 5, welche auf *M2* geplant war, wird ebenfalls auf Maschine *M3* umgeplant. Des Weiteren werden auch Operationen, welche zuvor nicht auf *M2* geplant waren und somit nicht direkt von dem Ausfall dieser Maschine betroffen sind, umgeplant. Begründen lässt sich dies dadurch, dass infolge der Neuplanung mehrerer Operationen von *M2* auf *M3* respektive *M4* umgeplant werden, wodurch sich eine Änderung der DLZ sowie eine Erhöhung der Auslastung dieser beiden Maschinen ergibt. Der Optimierungsalgorithmus plant, mit dem Ziel eine möglichst geringen DLZ zu erreichen, deshalb auch nicht direkt betroffene Operationen um. Der erstellte Ablaufplan verletzt keine der geltenden Randbedingungen und stellt somit einen gültigen Ablaufplan dar.

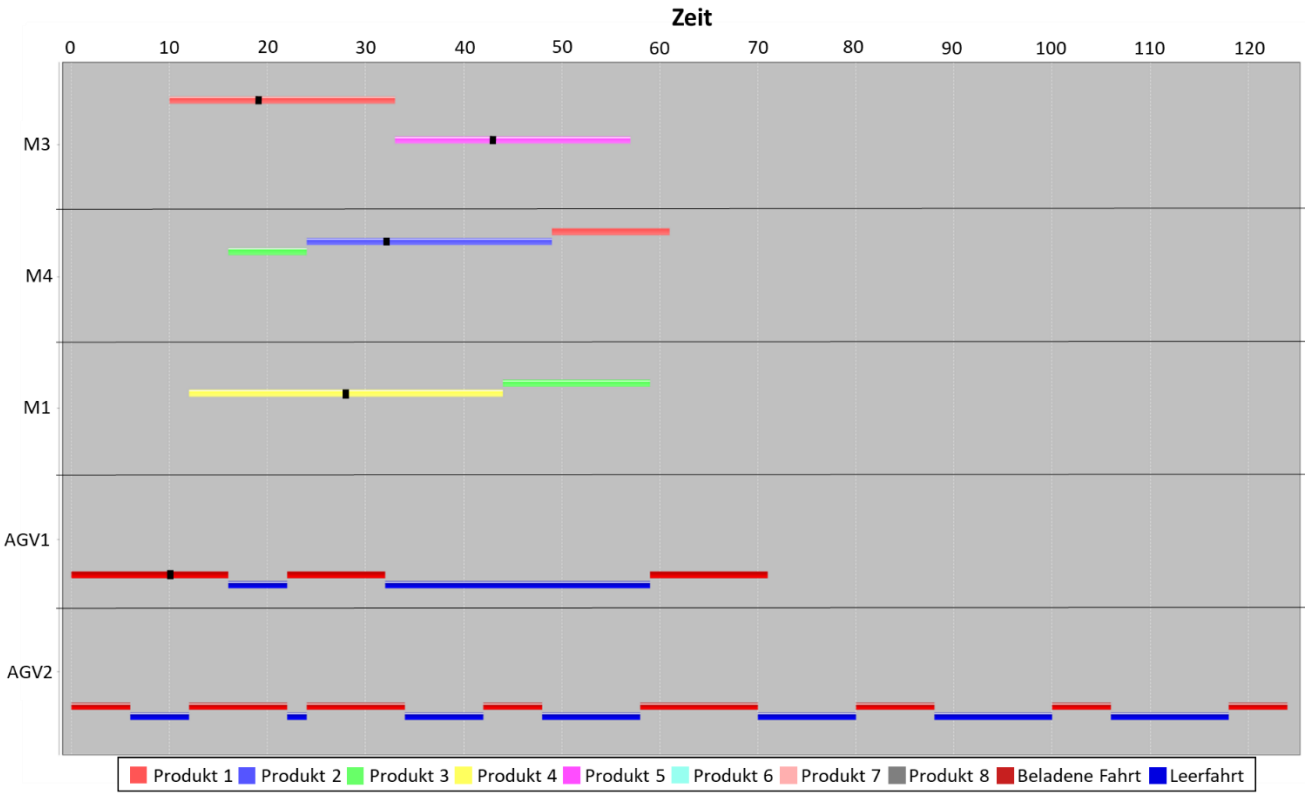


Abbildung 53: Neuer Ablaufplan nach Ausfall von M2 nach 15 ZE.

Der, als Reaktion auf den Ausfall eines der beiden FTS erstellte, Ablaufplan ist in Abbildung 54 zu sehen. Wie beim vorherig betrachteten Szenario ist die jeweils erste Operation von Produkt 2 und 3, aus den genannten Gründen, nicht mehr im Ablaufplan enthalten. Die sich ergebenden Verfügbarkeiten der Produkte 2 und 3 sowie die der bearbeitenden Maschinen werden entsprechend berücksichtigt. Alle Transportaufträge werden nach dem Ausfall von *AGV2* von *AGV1* übernommen. Der Optimierungsalgorithmus gestaltet den neuen Ablaufplan, durch die fehlende Transportkapazität so, dass möglichst wenige Transportaufträge zur Bearbeitung des Auftragssatzes notwendig sind. Alle noch durchzuführenden Operationen der meisten Produkte werden deshalb auf derselben Maschine bearbeitet. Diese ist auch der Fall für Operationen bei denen die Bearbeitungszeit auf einer anderen Maschine geringer wäre. Die Anzahl der notwendigen Transportaufträge verringert sich folglich, sodass trotz der reduzierten Transportkapazitäten eine möglichst geringe DLZ erreicht werden kann. Die Steuerungsarchitektur ist auch in dem Szenario in der Lage, keine der vorliegenden Randbedingungen zu verletzten und somit einen gültigen Ablaufplan zu generieren.

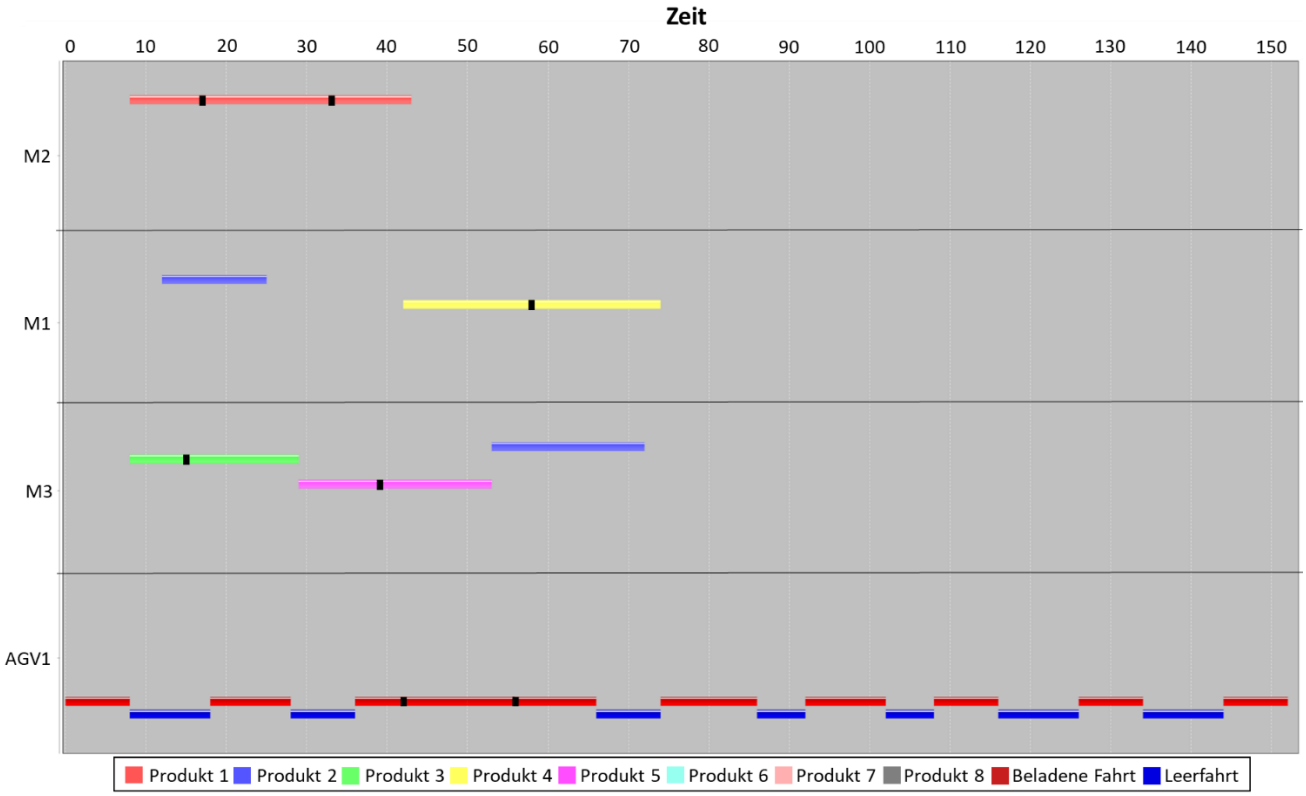


Abbildung 54: Neuer Ablaufplan nach Ausfall von AGV2 nach 15 ZE.

4.4.2 Reaktion auf neue, zu berücksichtigende Inspektionsergebnisse

Häufig ergeben sich in der Refabrikation, im Rahmen von Inspektionsprozessen, neue Erkenntnisse über den Zustand des Produktes. Dies kann die Durchführung einer zusätzlichen Operation notwendig machen. Auch in der Fertigung kann dies der Fall sein, wenn bspw. in einer Qualitätsprüfung Fehler festgestellt werden, die zur Nacharbeit des Produktes oder Bauteils führen. Eine Experten Komponente teilt der Steuerungsarchitektur in dieser Simulationsstudie nach 15 ZE eine zusätzlich durchzuführende Operation für Produkt 1 mit. Diese Operation soll auf Maschine *M1* bearbeitet werden und die Bearbeitungsdauer beträgt 10 ZE. Die neue Operation ist in der Operationsreihenfolge von Produkt 1 zwischen der zweiten und der ursprünglich dritten Operation einzuordnen.

Die Steuerungsarchitektur erzeugt für diese Szenario den in Abbildung 55 zu sehenden, neuen Ablaufplan. Wie bei den beiden vorherig betrachteten Fällen, ist die jeweils erste Operation von Produkt 2 und 3, aus den genannten Gründen, nicht mehr im Ablaufplan enthalten. Eine Berücksichtigung der entsprechenden Verfügbarkeiten erfolgt ebenfalls. Die zusätzliche Operation für Produkt 1 ist, wie vorgesehen, auf M1 mit eingeplant. Zudem wird die geforderte, neue Operationsreihenfolge für Produkt 1 eingehalten. Der erzeugte Ablaufplan verletzt keine der vorliegenden Randbedingungen, sodass die Steuerungsarchitektur auch in diesem Fall einen gültigen Ablaufplan erzeugt.

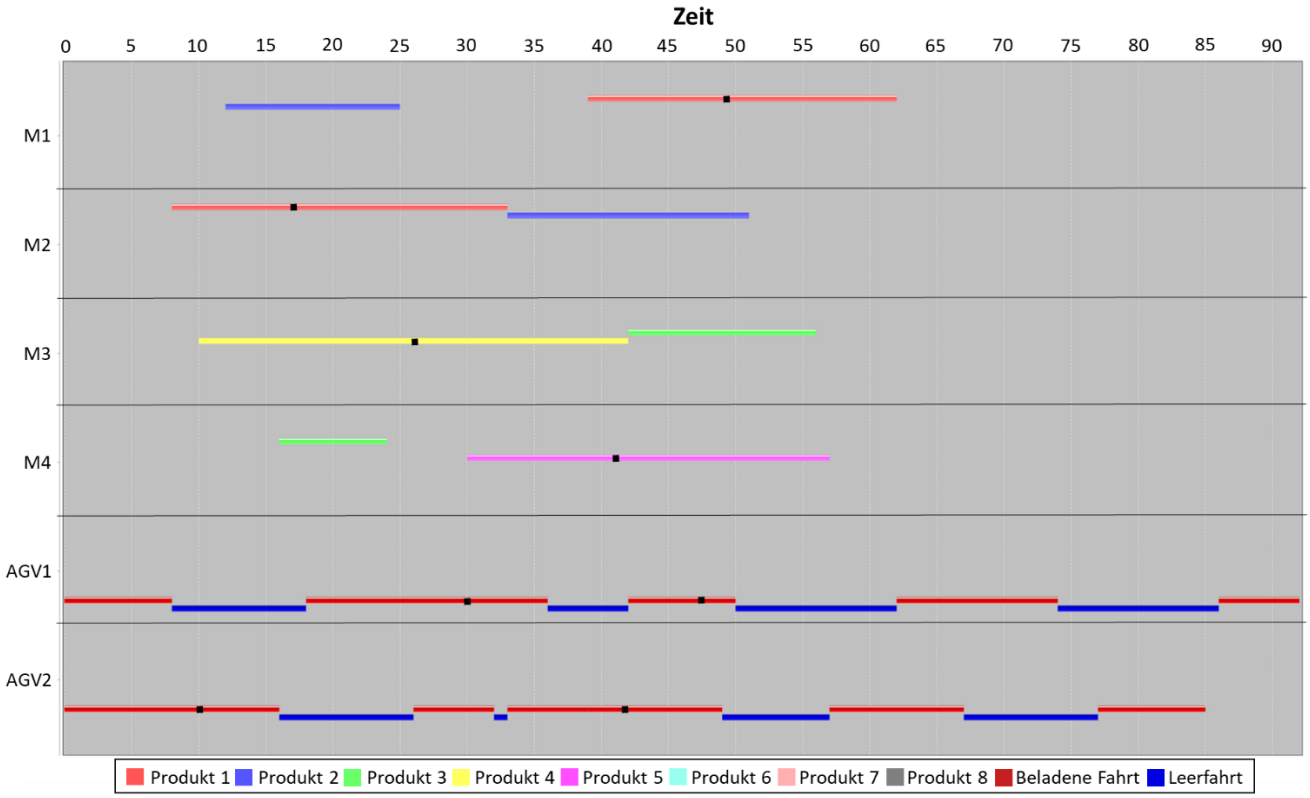


Abbildung 55: Neuer Ablaufplan nach zusätzlich hinzugekommener Operation M1(10) für Produkt 1 nach 15 ZE

4.4.3 Reaktion auf neuen, auszuführenden Auftrag

Ein in allen RS vorkommendes Ereignis ist das Eintreffen neuer, zu bearbeitender Aufträge. Die Ankunftszeit und die zur Bearbeitung des Auftrags notwendigen Operationen sind in der Refabrikation sowie bei der Fertigung kundenindividueller Produkte vorab nicht bekannt. Aus diesen Gründen handelt es sich auch hier um unerwartet, auftretende Ereignisse. Die neuen Aufträge sollen nicht erst in das RS eingeführt werden, wenn alle aktuell im Umlauf befindlichen Aufträge abgeschlossen sind, sondern kontinuierlich in den laufenden Prozess integriert werden. Die Integration eines neuen Auftrags in den aktuellen und sich in Ausführung befindlichen Ablaufplan, wird deshalb nachfolgend untersucht. Der zusätzliche Auftrag trifft nach 15 ZE ein und besteht aus den folgenden beiden Operationen, wobei der sich innerhalb der Klammern befindliche Wert, die Bearbeitungszeit angibt: *M3* (10), *M1* (21). Die Koordinator Komponente instanziiert und parametrisiert beim Eintreffen des neuen Auftrags eine zugehörige Produkt Komponente. Anschließend wird eine Neuplanung ausgelöst.

Der bei diesem Szenario neuerstellte Ablaufplan ist in Abbildung 56 dargestellt. Die jeweils erste Operation von Produkt 2 und 3 ist, aus den bereits genannten Gründen, nicht mehr im Ablaufplan enthalten. Zusätzlich in den Ablaufplan hinzugekommen ist der neue Auftrag (Produkt 6, türkisfarben dargestellt) mit der ersten Operation auf *M3* und der zweiten auf *M1*. Die Operationen des neuen Produktes werden nicht nach Abschluss der Bearbeitung aller im Umlauf befindlichen Produkte angehängt, wie es bspw. in Sahin et al. [168] der Fall ist, sondern in die laufende Prozessausführung integriert. Der neu erzeugte Ablaufplan verletzt keine der vorliegenden Randbedingungen, sodass ein gültiger Ablaufplan vorliegt.

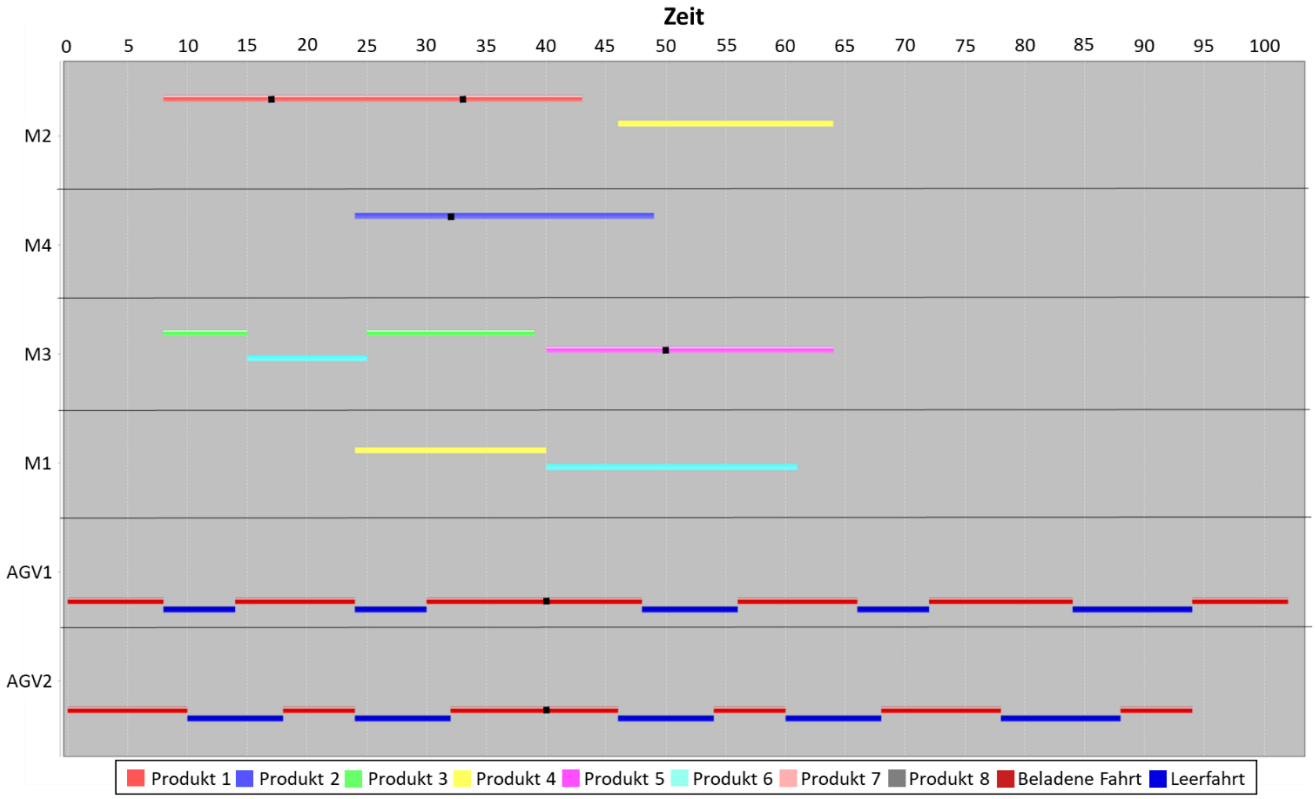


Abbildung 56: Neuer Ablaufplan nach dem Eintreffen eines neuen, zu bearbeitenden Auftrags (Auftrag 6) nach 15 ZE.

4.4.4 Reaktion auf nacheinander eintreffende, unvorhergesehene Ereignisse

Die vorherigen Simulationsstudien betrachten das separate Eintreten einzelner, unvorhergesehener Ereignisse. Diese treten in RS allerdings nicht nur getrennt voneinander, sondern auch parallel, oder nacheinander auf. Hierzu soll in diesem Kapitel die Reaktion der Steuerungsarchitektur auf mehrere, hintereinander auftretende, unerwartete Ereignisse untersucht werden. Auftragssatz 1 sowie Layout 1 der BI von Kumar et al. dienen auch in diesem Fall als Simulationsszenario. Die drei folgenden, nacheinander eintretenden Ereignisse werden untersucht:

1. Nach 15 ZE kommt eine neue Operation für Produkt 1, identisch zum Szenario in 4.4.2, hinzu
2. Nach weiteren 17 ZE kommt ein neuer Auftrag *M3* (10), *M1* (21) hinzu
3. Nach weiteren 20 ZE fällt das FTS *AGV1* aus

Die Steuerungsarchitektur erzeugt nach dem Eintreten des ersten Ereignisse den in Abbildung 55 dargestellten Ablaufplan. Das zweite Ereignis tritt 17 ZE nach dem Start der Ausführung dieses Ablaufplans ein. Die Steuerungsarchitektur erzeugt als Reaktion darauf den in Abbildung 57 dargestellten Ablaufplan. Die beide Operationen des neu hinzugekommenen Auftrags (Produkt 6, türkis) sind darin korrekt eingeplant. Die beiden Operationen werden nicht nach der vollständigen Bearbeitung der initial vorhandenen Aufträge dem Ablaufplan angehängt, sondern in diesen vollständig integriert. Des Weiteren ist festzustellen, dass die beiden, sich zum Zeitpunkt der Neuplanung in Bearbeitung befindlichen Operationen der Produkte *1* (zwei Operationen), *2*, *3* und *4*, sich nicht mehr im Ablaufplan befinden. Der Ablaufplan in Abbildung 57 verletzt keine der geltenden Randbedingungen und ist somit gültig.

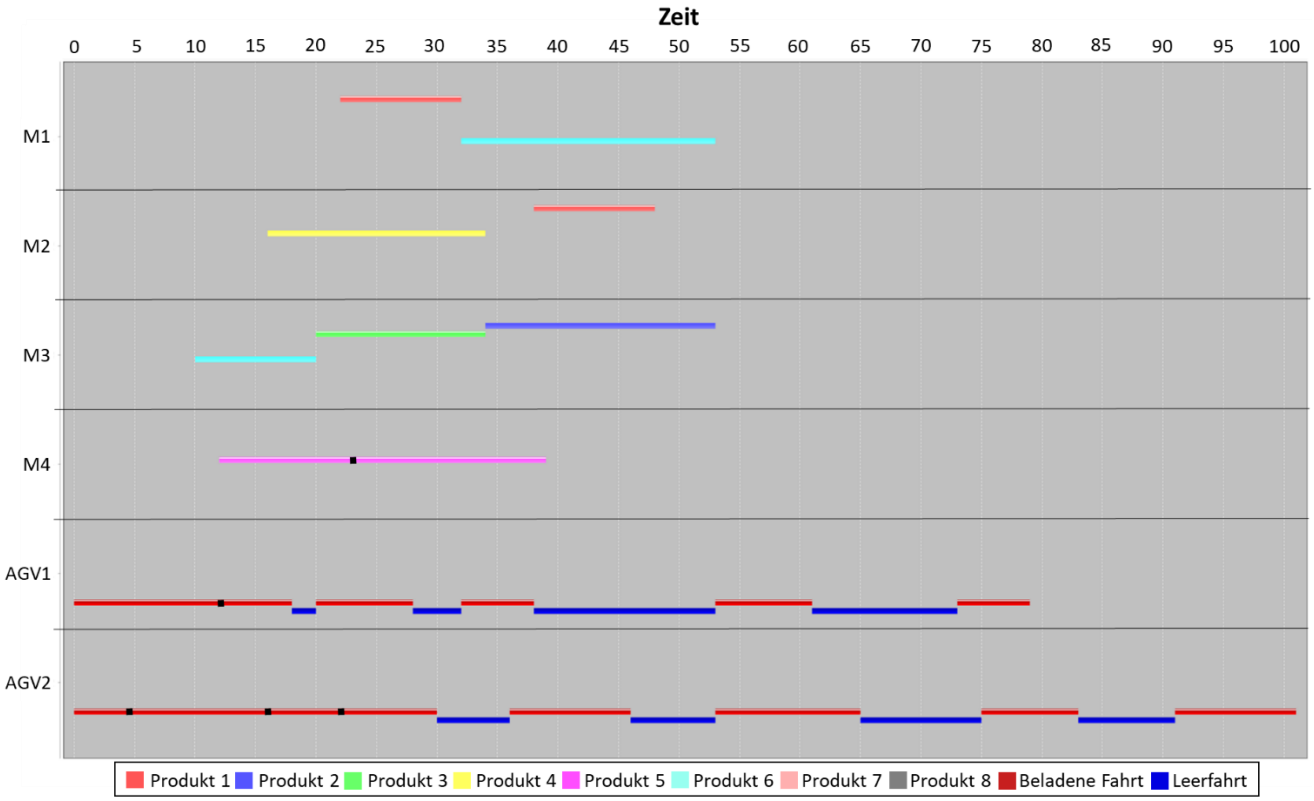


Abbildung 57: Neuer Ablaufplan nach dem Eintritt des zweiten unerwarteten Ereignisses.

Die Steuerungsarchitektur erstellt, als Reaktion auf das Eintreten von Ereignis 3, den in Abbildung 58 dargestellten Ablaufplan. *AGV1* befindet sich zum Eintrittszeitpunkt dieses Ereignisses in einer Leerfahrt und *AGV2* in einer Transportfahrt. Produkt *1* wird bei dieser zu Maschine *M1* transportiert (siehe Abbildung 57). *AGV2* muss, bezogen auf den Zeitpunkt der Neuplanung, weitere 2 ZE mit der Fertigstellung dieser Transportfahrt verbringen, bevor es mit der Ausführung des neuen Ablaufplans beginnen kann. Zudem wird bei der Neuplanung berücksichtigt, dass *AGV2* diese Fahrt an *M1* beendet und somit evtl. eine Leerfahrt zum Aufnehmen des nächsten Produktes durchführen muss. Diese Leerfahrt ist in Abbildung 58 nicht als eigener Block aufgeführt, wird aber, wie zu sehen, für den Startzeitpunkt der nächsten beladenen Fahrt mit berücksichtigt. *AGV2* transportiert bei dieser Produkt 6 von *M3* auf *M1* und muss somit zuerst eine 6 ZE dauernde Leerfahrt zwischen seiner aktuellen Position (*M1*) und *M3* durchführen. Hierdurch ergibt sich der zu sehende Startzeitpunkt der beladenen Fahrt bei 8 ZE. Operationen der Produkte *4*, *5* und *6* befinden sich zu diesem Zeitpunkt auf den Maschinen *M2*, *M4* und *M3* in Bearbeitung. Die Bearbeitung der ersten Operation von Produkt *6* wird zum Zeitpunkt der Neuplanung gerade fertiggestellt und die Bearbeitung der letzten Operation von Produkt *3* anschließend auf derselben Maschine (*M3*) gestartet. Alle sich daraus ergebenden Zustände und Verfügbarkeiten der Produkte, Maschinen und FTS wurden bei der Neuplanung berücksichtigt. Nach dem Ausfall von *AGV1* ist, wie in dem neuen Ablaufplan zu sehen, nur noch *AGV2* als Transportmittel vorhanden. Dieses führt alle noch vorhandenen Transportaufträge aus. Transportaufträge, die zuvor *AGV1* zugeteilt waren, werden im Rahmen der Neuplanung auf *AGV2* umgeplant. Weitere vier Transportfahrten sind nach Abschluss der Bearbeitung der letzten Operationen zu sehen. Bei diesen beladenen Fahrten handelt es sich um den Rücktransport der fertigbearbeiteten Aufträge in ein Lager (*L/U*-Station innerhalb der BI von Kumar et al.).

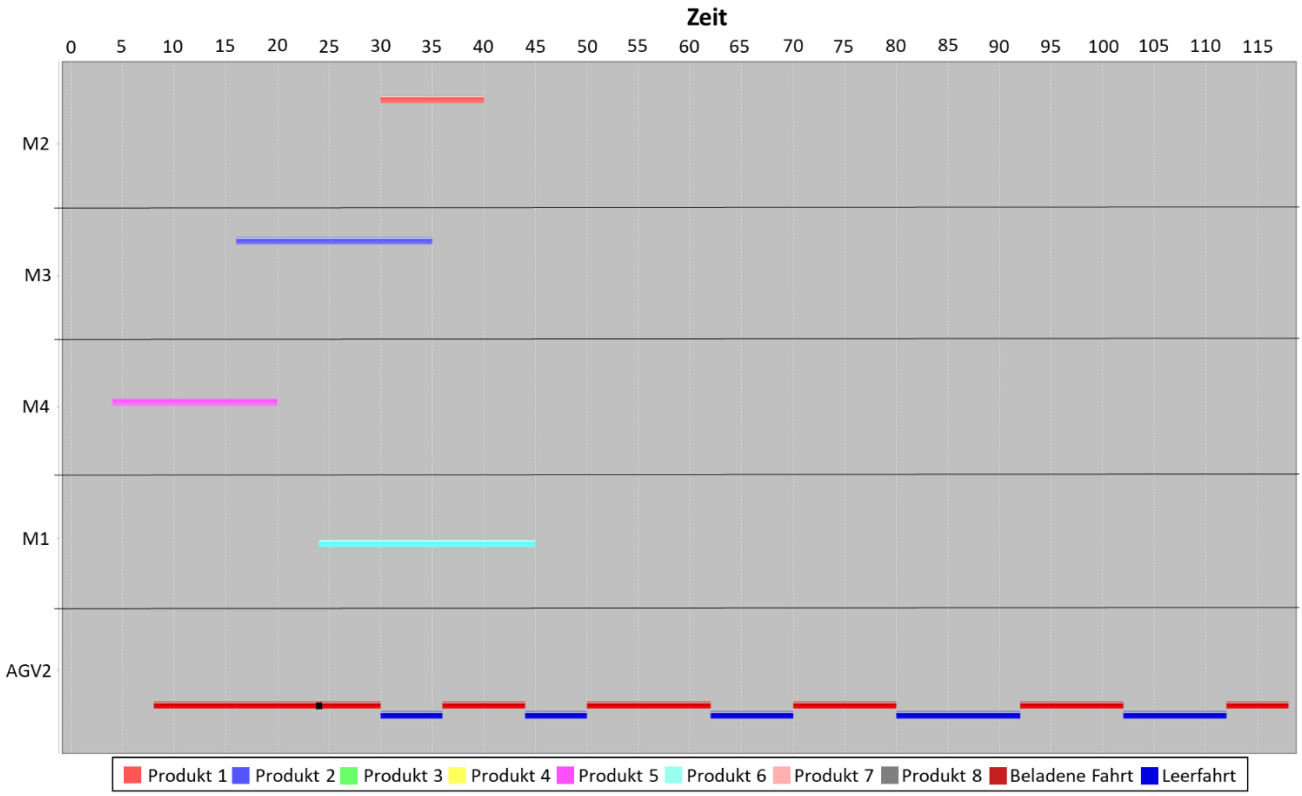


Abbildung 58: Neuer Ablaufplan nach dem Eintritt des dritten unerwarteten Ereignisses.

Die 3. Forschungsfrage *„Kann eine hybride Steuerungsarchitektur adäquat auf unerwartet auftretende Ereignisse reagieren, und gleichzeitig das Gesamtsystem global optimieren?“* kann durch die Ergebnisse dieses Kapitels positiv beantwortet werden. Die Simulationsergebnisse in Kapitel 4.3 zeigen, dass der vorgestellte Ansatz gleich gute und oftmals bessere Ergebnisse bzgl. der DLZ-Minimierung liefert, als die zum Vergleich herangezogenen Optimierungsverfahren für die SAMF. Diese trifft sowohl gegenüber den herangezogenen zentralen als auch dezentralen, selbstorganisierenden Ansätze zu. Eine Optimierung des Gesamtsystems kann damit bestätigt werden. Die Simulationsergebnisse in Kapitel 4.4 zeigen, dass die hybride Steuerungsarchitektur zudem in der Lage ist adäquat auf unerwartet auftretende Ereignisse zu reagieren.

4.5 Zwischenfazit

Die zur SAMF entwickelten Optimierungsverfahren und CP-Modelle wurden in diesem Kapitel vorgestellt. Darüber hinaus wurde die Interaktion zwischen den einzelnen Komponenten der hybriden Steuerungsarchitektur, zur Durchführung einer Neuplanung beim Auftreten unerwarteter Ereignisse, dargestellt. Abschließend wurden die entwickelten Verfahren im Rahmen verschiedener Simulationsstudien untersucht und verifiziert. Verschiedene Benchmarks wurden hierzu genutzt.

In der ersten Simulationsstudie wurde untersucht, ob die Verwendung von LNS oder GA bessere Ergebnisse bzgl. der Optimierung der CP-Modelle liefert. Die LNS erzielt hierbei bessere Ergebnisse bzgl. DLZ und dafür notwendiger RZ. Eine generelle Gegenüberstellung der simultanen, sequenziellen und selbstorganisierten Ablaufplanung erfolgte anschließend. Die SAMF erzielt hierbei eine deutliche DLZ-Reduzierung gegenüber der sequenziellen (35,6 %), als auch der selbstorganisierten (40,3 %) Ablaufplanung. Das RS gewinnt durch diese zielführende Integration der FTS-Steuerung in die Ablaufplanung einen wesentlich höheren Durchfluss sowie eine deutlich höhere Produktionskapazität.

Der in dieser Arbeit erfolgreich entwickelte CP-basierte Ansatz wurde des Weiteren, anhand von standardisierten BI bzgl. DLZ und RZ, mit den Ergebnissen anderer Veröffentlichungen zur SAMF verglichen. Dieser kann bzgl. beider Kriterien mindestens gleich gute und meist bessere Ergebnisse, als die zum Vergleich herangezogenen, Ansätze liefern. Besonders die geringe und in diesem Vergleich geringste, RZ macht die sehr gute Eignung des vorgestellten Ansatzes zur Steuerung realer RS deutlich. Die Nutzung verschiedener sowie lexikografischer, multikriterieller Zielfunktionen innerhalb des CP-basierten Ansatzes wurde ebenfalls untersucht. Die Auswirkung der genutzten Zielfunktion auf diverse KPIs wurde dabei betrachtet.

Nach diesen rein deterministischen Untersuchungen wurde die Reaktion der Steuerungsarchitektur auf verschiedene, unerwartet auftretende Ergebnisse simuliert und untersucht. Diese ist in der Lage, auf alle simulierten Ereignisse schnell und adäquat zu reagieren. Während jeder Neuplanung konnte innerhalb einer Sekunde ein neuer, gültiger Ablaufplan erzeugt werden. Die Kombination der vorgestellten Neuplanungs-Strategie der hybriden Steuerungsarchitektur mit dem entwickelten CP-Modell zur SAMF, ermöglicht es zudem, die Nachteile der häufig zur Neuplanung genutzten reinen Konstruktionsverfahren zu eliminieren. Diese liefern zwar in kurzer Zeit an die aktuelle Situation angepasste und gültige Ablaufpläne, allerdings erreichen die erstellten Ablaufpläne lediglich eine schlechte Optimierungsqualität. Das vorgestellte Verfahren bietet eine wesentlich bessere Optimierungsqualität im Vergleich zu einfachen Konstruktionsverfahren. Das Verfahren kann diese dennoch in ausreichend kurzer RZ liefern, wodurch es der Lage ist schnell auf unerwartet auftretende Ereignisse zu reagieren. Dies qualifiziert die Steuerungsarchitektur für den generellen Einsatz in realen RS.

5 Validierung der Steuerungsarchitektur

Neben der Verifizierung der vorgestellten Steuerungsarchitektur im Rahmen von Simulationsstudien, erfolgt die Implementierung dieser in einer smarten Modellfabrik, mit dem Ziel diese zu steuern. Die Steuerungsarchitektur soll in der Lage sein, alle relevanten Daten der Fertigungsteilnehmer in Form eines Echtzeit-Feedbacks zu sammeln. Sie ist dadurch beim Auftreten unerwarteter Ereignisse, in der Lage eine Neuplanung, unter Berücksichtigung des aktuellen Zustands der Fertigungsebene, autonom durchzuführen. Eine erfolgreiche Umsetzung dieses Vorhabens kann als Validierung der grundsätzlichen Anwendbarkeit des Ansatzes zur Steuerung realer RS angesehen werden. Die Implementierung des Ansatzes innerhalb einer Modellfabrik hat gegenüber der Implementierung in einer industriellen Fertigungsumgebung den Vorteil, dass die Funktion und das Verhalten des Systems in verschiedenen Situationen getestet werden kann, ohne Produktionsstillstände zu verursachen. Produktionsstillstände sind mit hohen Kosten durch geringere, produzierte Stückzahlen, Unterbrechungen in der Supply-Chain und Konventionalstrafen beim nicht einhalten von Lieferterminen verbunden und somit wirtschaftlich nicht vertretbar. Die erfolgreiche Implementierung erhöht darüber hinaus die Akzeptanz zum Testen der entwickelten Steuerungsarchitektur in einer industriellen Umgebung bei potenziellen Nutzern.

Die Bezeichnung der Modellfabrik als smart ist eine Anlehnung an die Begrifflichkeit der *Smart Factory* (Intelligente Fabrik), die innerhalb der *Umsetzungsempfehlung für das Zukunftsprojekt Industrie 4.0* [6] wie folgt definiert wird: *„Einzelnes oder Verbund von Unternehmen, das / der IKT zur Produktentwicklung, zum Engineering des Produktionssystems, zur Produktion, Logistik und Koordination der Schnittstellen zu den Kunden nutzt, um flexibler auf Anfragen reagieren zu können. Die Smart Factory beherrscht Komplexität, ist weniger störanfällig und steigert die Effizienz in der Produktion. In der Smart Factory kommunizieren Menschen, Maschinen und Ressourcen selbstverständlich miteinander wie in einem sozialen Netzwerk.“* [6]. Die hybride Steuerungsarchitektur vernetzt die einzelnen Fertigungsteilnehmer wie Maschinen, FTS und Produkte zu einem CPS (siehe Abbildung 59). Dadurch wird, laut der Definition von Geisberger et al. [187], aus dem RS automatisch eine *Smart Factory*. In [187] ist der Zusammenhang zwischen CPS und *Smart Factory* wie folgt definiert: „*Der Einsatz von Cyber-Physical Systems in Produktionssystemen führt zur, Smart Factory‘.“* [187]. Die Steuerungsarchitektur verwandelt auch die innerhalb des RS befindlichen Produkte in smarte Produkte, denn sie *„sind eindeutig identifizierbar, jederzeit lokalisierbar und kennen ihre Historie, ihren aktuellen Zustand sowie alternative Wege zum Zielzustand.“* [6].

In diesem Kapitel erfolgt:

- die Vorstellung der smarten Modellfabrik mit ihren einzelnen Fertigungsteilnehmern,
- die Umsetzung der Vernetzung zwischen den Teilnehmern,
- die Integration der vorgestellten Steuerungsarchitektur in die Modellfabrik sowie
- die Beschreibung des untersuchten Use Cases und der erzielten Ergebnisse.

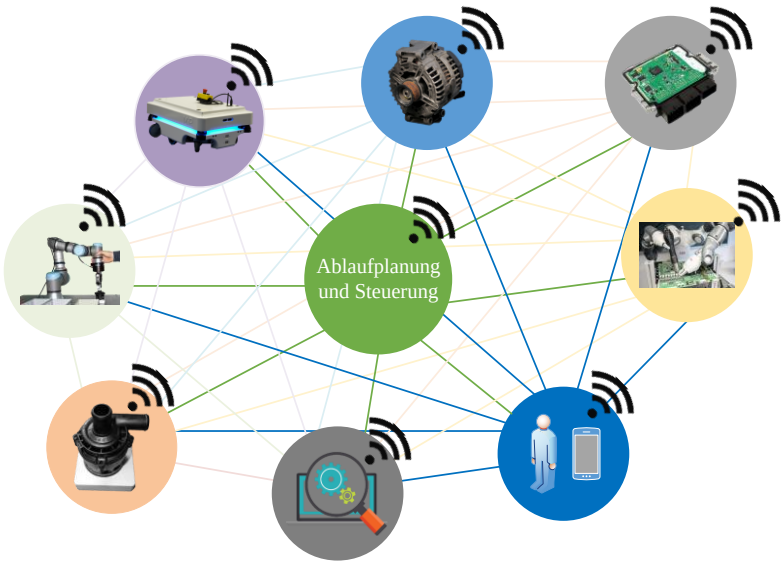


Abbildung 59: Vernetzte Fertigungsteilnehmer zu einem CPS.

5.1 Aufbau der smarten Modellfabrik zur Refabrikation

Innerhalb der Modellfabrik sollen zum einen ein industrienaher Use Case dargestellt, und zum anderen die verschiedenen Fähigkeiten der hybriden Steuerungsarchitektur demonstriert werden. Diese umfassen u. a.:

- SAMF
- Abbildung alternativer Ressourcen (FJSSP)
- Erstellung eines neuen, angepassten Ablaufplans beim Auftreten unvorhergesehener Ereignisse
- Nutzung von Echtzeitdaten aus der Fertigungsebene, um jederzeit ein aktuelles Abbild dieser zu haben
- Steuerung unterschiedlicher Fertigungsteilnehmer wie Roboter, Maschinen, manuelle Arbeitsplätze und FTS
- Nutzung des aktuellen Status sowie der Verfügbarkeit aller relevanten Fertigungsteilnehmer bei der Neuplanung
- Anpassung des Ablaufplans unter Berücksichtigung der Inspektionsergebnisse eines zu refabrizierenden Produktes
- Vernetzung aller Fertigungsteilnehmer zu einem CPS durch den Einsatz von *Internet of Things (IoT)* und *Embedded Systems*

Die Modellfabrik ist im Labor des Fachgebiets Robotik und Regelungstechnik am Umwelt-Campus Birkenfeld der Hochschule Trier aufgebaut. Innerhalb der Modellfabrik sind die folgenden Stationen vorhanden:

- Lagerstation
- Demontagestation 1*(Ressourcen Komponente)*
- Demontagestation 2 *(Ressourcen Komponente)*
- Inspektionsstation (*Experten Komponente*)
- Reinigungsstation *(Ressourcen Komponente)*
- Aufbereitungsstation *(Ressourcen Komponente)*

- Montagestation *(Ressourcen Komponente)*
- FTS (*Transport Komponente*)

Das Layout der Modellfabrik mit den einzelnen, zuvor aufgelisteten Stationen sowie deren räumliche Anordnung ist in Abbildung 60 dargestellt.

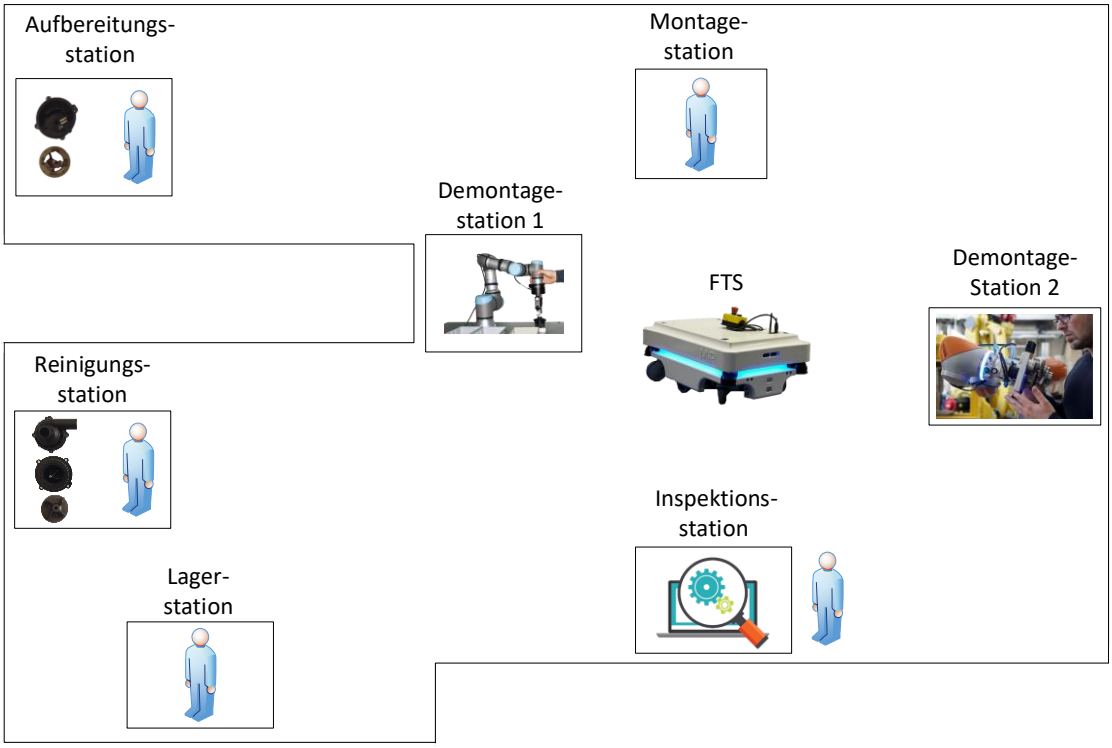


Abbildung 60: Layout der Modellfabrik

Das Modell *MiR100* der Firma *Mobile Industrial Robots* wird als FTS verwendet (Abbildung 61). Eine Hubeinheit für das Auf- und Abladen der Transportboxen an den einzelnen Stationen bzw. Pufferplätzen ist auf dieser angebracht. Des Weiteren verfügt sie über Zentrierhilfen, welche die Positionsungenauigkeit des FTS ausgleichen. Die Box zentriert sich beim Absetzen dieser auf dem FTS durch die Geometrie der Zentrierhilfen selbst. Die *MiR100* kann ein Produkt zwischen zwei beliebigen Stationen transportieren und ist nicht auf vordefinierte Pfade angewiesen. Sie ist in der Lage diese, unter Berücksichtigung seiner Umgebung, selbst zu planen.

Abbildung 61: FTS *MiR100* der Firma Mobile Industrial Robots

Die 2D-Karte der Modellfabrik, welche die MiR100 als Navigations- und Lokalisierungsgrundlage nutzt, ist in Abbildung 62 dargestellt. Alle Anfahrpunkte des FTS zum Auf- und Abladen der Produkte an den einzelnen Stationen werden darin als blaue Kreise dargestellt. Schwarze Linien stellen den Grundriss der Modellfabrik dar und rote Linien sind alle Konturen, welche die MiR100 zum aktuellen Zeitpunkt durch ihre Sensorik erfassen kann. Die in Abbildung 62 zu sehende 2D-Karte entspricht dem konzeptionellen Layout der Modellfabrik in Abbildung 60.

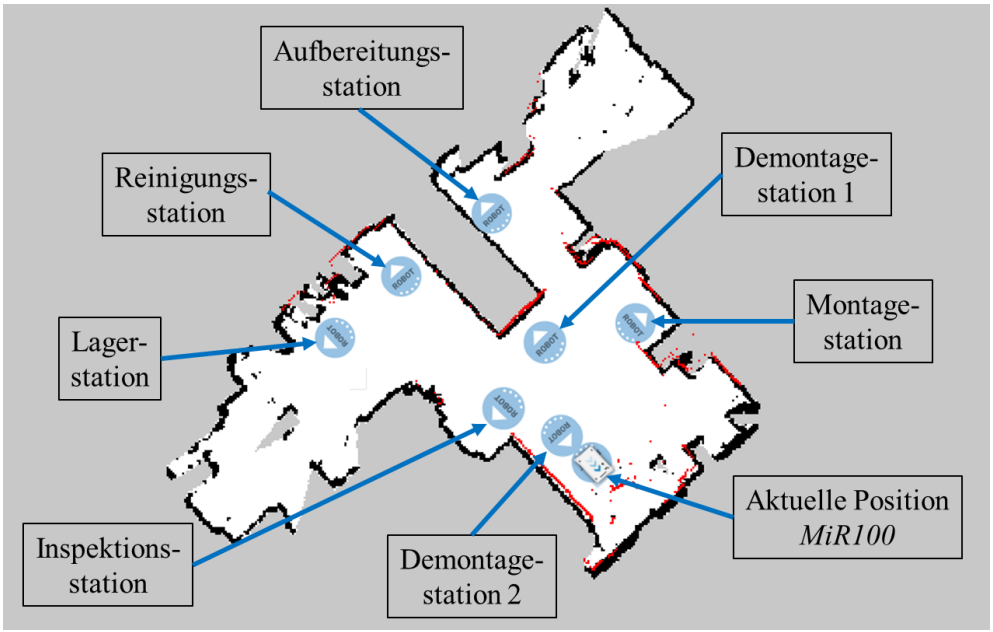


Abbildung 62: 2D-Karte der Modellfabrik zur Navigation und Lokalisierung der MiR100.

Die Verwendung von zwei Demontagestationen ermöglicht es, alternative Maschinen (FJSSP) innerhalb der Ablaufplanung darzustellen. Die beiden Demontagestationen sind jeweils in der Lage das vorliegende Produkt zu demontieren. Sie sind allerdings nicht identisch. Demontagestation 1 besteht aus einem *Universal Robots UR3* Roboter und Demontagestation 2 aus einem *KUKA LBR iiwa* Roboter (siehe Abbildung 63). Die Prozesszeiten der Demontage eines Produktes unterscheiden sich in Abhängigkeit der bearbeitenden Demontagestation.

Abbildung 63: Alternative Demontagestationen 1 und 2 (Vordergrund: Universal Robots UR3; Hintergrund: KUKA iiwa) mit Transportbox (grün) und RFID-Reader (rote Box) zur eindeutigen Identifikation des vorliegenden Produktes.

In die Ressourcen Komponente des *KUKA LBR iiwa* wird der von Jungbluth et al. [178] entwickelte *intelligente Demontage-Assistent-Agent (IDAA)* integriert. IDAA ist ein auf MAS-Technologie basierendes System zur automatisierten Erzeugung und Ausführung von Demontageplänen. Es ist in der Lage Assistenzrobotersysteme für die Ausführung des erzeugten Demontageplans zu steuern. Die Demontageplanung erfolgt basierend auf Produktmodellen. Diese werden ausgehend von den CAD-Daten der Produkte sowie an diesen zusätzlich angehängten Demontageinformationen erzeugt. Produktmodelle werden in der dezentralen Wissensbasis der Steuerungsarchitektur gespeichert (siehe Abbildung 25) und durch die entsprechende Ressourcen Komponente IDAA zur Verfügung gestellt. Das Produktmodell der als Use Case genutzten Zusatzkühlmittelpumpe ist in Abbildung 64 als Produktgraph dargestellt. Die einzelnen Blöcke repräsentieren die Bauteile der Pumpe, und die Pfeile (Kanten) die zwischen den Bauteilen bestehenden Verbindungen. Nur Bauteile, die keine eingehenden Kanten haben, können demontiert werden. Hat ein Bauteil eingehende Kanten, muss zuerst das Bauteil demontiert werden, von dem diese Kanten ausgehen. Für eine detailliertere Erläuterung des Aufbaus und der Funktionsweise von IDAA wird auf Jungbluth, Gerke und Plapper [178] verwiesen.

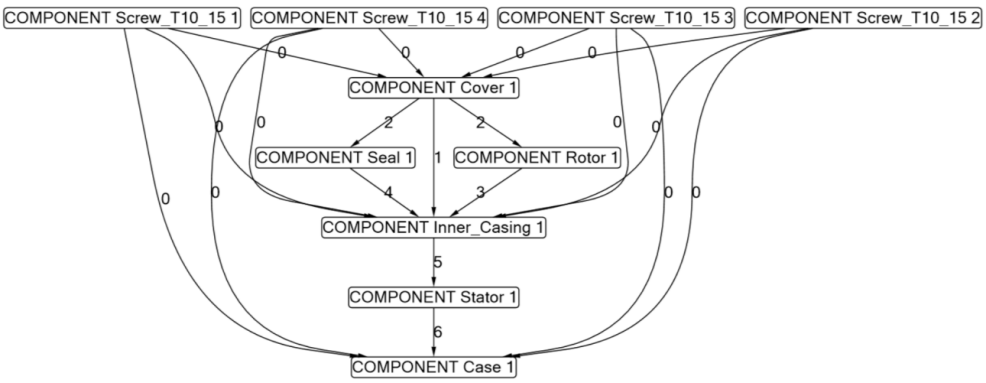


Abbildung 64: Produktmodell(-graph) für die verwendete Kühlmittelpumpe.

Der beispielhafte Aufbau einer Station (Demontagestation 1) mit den zugehörigen Komponenten ist in Abbildung 65 dargestellt. Die Station besteht aus folgenden Komponenten:

- Universal Robots UR3 inkl. Kamera- und Greifsystem
- Pufferplatz 1 und 2
- RFID-Reader an jedem Pufferplatz

Des Weiteren ist in der Abbildung das FTS *MiR100* und die darauf montierte Hubeinheit zu sehen. Ein RFID-Chip ist an jeder Transportbox angebracht. Das in der Box befindliche Produkt lässt sich über diesen eindeutig identifizieren. Der an der Station befindliche RFID-Reader ermöglicht, über das Auslesen des RFID-Chips, zum einen die Identifikation des vorliegenden Produktes und zum anderen eine Anwesenheitskontrolle für den jeweiligen Pufferplatz. Solange der RFID-Reader einen RFID-Chip erkennt, wird davon ausgegangen, dass sich eine Transportbox auf dem Pufferplatz befindet. Der RFID-Reader ist mit der Ressourcen Komponente der Station verbunden. Die Ressourcen Komponente führt nach der Identifikation einer neu angekommenen Transportbox die Bestimmung des zugehörigen Prozesses durch und startet dessen Ausführung anschließend. Das FTS startet, nach dem Abladen einer Transportbox an einem Pufferplatz, seinen nächsten Transportauftrag. Die zugehörige Transport Komponente übermittelt diesen Auftrag an das FTS.

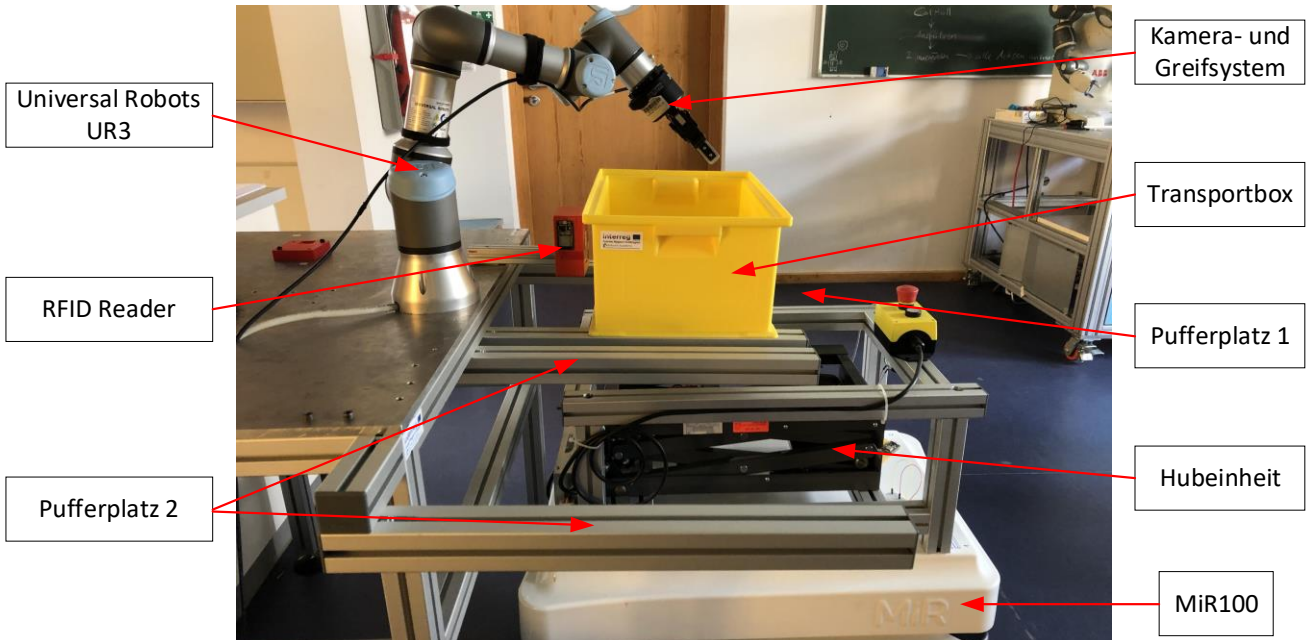


Abbildung 65: Anlieferung eines Produktes an Demontagestation 1

Die Steuerungsarchitektur muss zur Durchführung der SAMF die Fahrzeiten des verwendeten FTS zwischen den einzelnen Stationen der Modellfabrik kennen. Diese sind in der Transportzeitenmatrix in Tabelle 14 aufgeführt. Die Transportzeiten umfassen nicht nur die reine Fahrzeit zwischen zwei Stationen, sondern berücksichtigen auch die Zeit, die jeweils für das Auf- und Abladen der Transportboxen an den Stationen benötigt wird. Auf- und Abladezeiten sind somit in den Transport- und nicht in den Prozesszeiten enthalten. Grund hierfür ist, dass das Auf- und Abladen zwar das FTS zeitlich bindet, allerdings nicht die bearbeitende Fertigungsressource. Diese ist währenddessen zur Bearbeitung anderer Operationen verfügbar. Die einzelnen, in der Tabelle eingetragenen Werte, sind direkt von der Steuerungssoftware des FTS ausgelesen, da dieses jede Fahrt zeitlich mitverfolgt und so genaue Auskunft über die Fahrzeit zwischen zwei Positionen gibt. Für die manuellen und automatischen Auf- und Abladevorgänge wird der Mittelwert über jeweils zehn Vorgänge herangezogen. An jeder Station sind mehrere Pufferplätze vorhanden, sodass jeweils mehrere Transportboxen gelagert werden können.

		Zu [sec]						
	Z / V	Lager	Demont. 1	Demont. 2	Inspektion	Reinigung	Aufbereitung	Montage
Von [sec]	Lager	-	75	74	75	71	74	77
	Demont. 1	54	-	73	59	64	55	59
	Demont. 2	63	63	-	68	68	63	73
	Inspektion	55	66	69	-	65	55	64
	Reinigung	48	75	73	74	-	70	75
	Aufbereitung	72	77	78	85	82	-	72
	Montage	57	61	68	60	66	53	-

Tabelle 14: Transportzeitenmatrix zwischen den einzelnen Stationen innerhalb der Modellfabrik inkl. der Auf- und Abladevorgänge der Transportboxen.

Die Durchführung des Hubes für den Abladevorgang entfällt bei Leerfahrten, sodass sich die Fahrzeit bei diesen um 14 sec, gegenüber den beladenen Fahrten, reduziert. Eine zusätzliche Leerfahrtenzeitenmatrix, zur Parametrisierung der Leerfahrten innerhalb der SAMF Berücksichtigung, ist deshalb notwendig. Die Leerfahrtenmatrix für die Modellfabrik ist im Anhang R zu finden.

In dieser sind darüber hinaus die Zeiten enthalten die das FTS benötigt, um nach dem Abladen einer Transportbox an einem Pufferplatz eine Transportbox von einem anderen Pufferplatz derselben Station aufzunehmen. Solche Leerfahrten sind in der Diagonalen der Leerfahrtenmatrix enthalten. Diese Fahrzeiten setzen sich aus der Fahrzeit für den Rangiervorgang zwischen den Pufferplätzen sowie der Zeit zum Aufnehmen einer Transportbox zusammen. Diese Zusammensetzung der Fahrzeiten gilt für die Demontagestationen 1 und 2 sowie die Montagestation, jedoch nicht für die weiteren Stationen, da bei diesen das Be- und Entladen des FTS manuell geschieht. Das FTS verfährt an diesen Stationen nicht zwischen unterschiedlichen Pufferplätzen, sondern bleibt an einer Position für den Auf- und Abladevorgang stehen. Der Werker platziert, nachdem er die zuvor gelieferte Transportbox von dem FTS entnommen hat, die neue Transportbox auf dem FTS.

Der Fertigungssteurer startet die Modellfabrik über die in Abbildung 66 dargestellte GUI. Diese ist dient als Kommunikationsschnittstelle zwischen dem Fertigungssteuerer und der Koordinator Komponente. Über die Schaltfläche *Ablaufplanung starten* wird ein Satz an zu refabrizierenden Produkten initialisiert und die Scheduler Komponente beginnt mit der Erstellung des initialen Ablaufplans. Eine zugehörige Produkt Komponente wird für jedes Produkt instanziiert. Die Scheduler Komponente übermittelt diesen den zugehörigen Teil des Ablaufplans. Dieser wird nach Fertigstellung in Form eines Gantt-Diagramms dargestellt. Die Transport Komponente erhält ebenfalls den für sie notwendigen Teil des Ablaufplans. Der Fertigungssteurer kann über die Schaltfläche *Refabrikation starten* diese anstoßen. Das FTS beginnt mit der Aufnahme des ersten Produktes, sobald der zuvor beschriebene Prozess abgeschlossen ist. Die Steuerungsarchitektur führt anschließend alle notwendigen Prozesse zur Ausführung dieses Ablaufplans durch. Im rechten Teil der GUI kann die Anzahl der verfügbaren FTS definiert sowie eine Abschaltung von Demontagestation 1 (UR3) und 2 (iiwa) der Steuerungsarchitektur mitgeteilt werden. Diese führt in allen drei Fällen eine Neuplanung durch. Die manuelle Initiierung einer Neuplanung ist auch über die Schaltfläche *Neuplanung starten* möglich. Der obere, rechte Teile der GUI dient der Definition und dem Anlegen eines neuen Auftrags, der in den laufenden Betrieb der Modellfabrik integriert werden soll. Es können fünf Operationen (maximale Anzahl an Prozessen innerhalb der Modellfabrik) für den Auftrag definiert werden. In die Felder der Spalte *M1* kann der Index der ausführenden Station und in *P1* die zugehörige Prozesszeit eingetragen werden. Die Felder der Spalten *M2* und *P2* dienen der Definition einer alternativen Station zur Durchführung der jeweiligen

Operation. Die Operationsreihenfolge ist mit der Reihenfolge der einzelnen Zeilen deckungsgleich. Der Fertigungssteuer kann über die Schaltfläche *Neuen Auftrag anlegen* die Berücksichtigung des neuen, zu refabrizierenden Produktes anstoßen. Für jedes neu angelegte Produkt wird hierzu eine Produkt Komponente instanziiert und mit den zu bearbeitenden Operationen sowie deren Reihenfolge parametrisiert. Der Koordinator Komponente wird das Anlegen eines neuen Produktes mitgeteilt. Diese initiiert daraufhin eine Neuplanung, um die Bearbeitung des neuen Produktes in den laufenden Prozess zu integrieren.

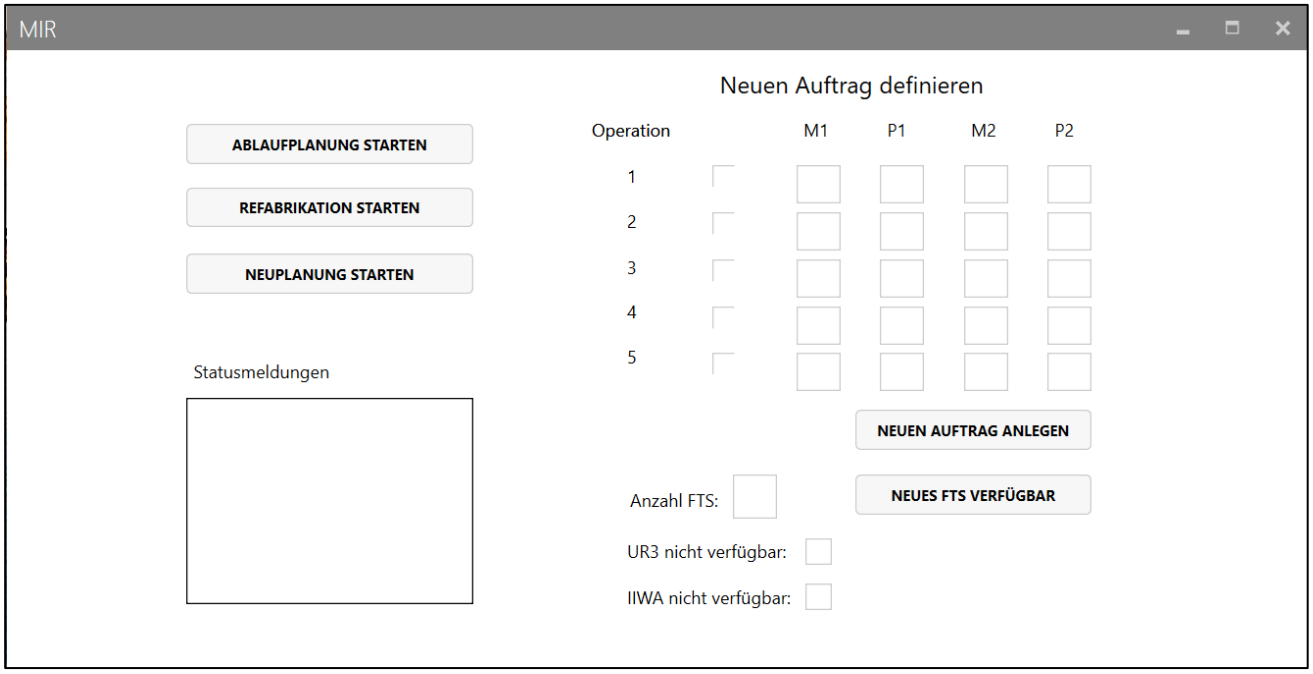


Abbildung 66: Graphische Benutzeroberfläche des Fertigungssteurers für die Modellfabrik.

5.1.1 Vernetzung der einzelnen Fertigungsteilnehmer

Die physischen Fertigungsteilnehmer müssen, zur Realisierung der hybriden Steuerungsarchitektur, mit den zugehörigen, softwaretechnischen Komponenten der Architektur verbunden sein. Die Komponenten haben die Aufgabe, die Fertigungsteilnehmer zu steuern, Prozessdaten von diesen auszulesen sowie untereinander zu kommunizieren. Die beiden Demontagestationen, die Montage-, Reinigungs-, Inspektions-, Aufbereitungs- und Lagerstation sind hierzu mit eingebetteten Systemen versehen. Auf diesen sind die entsprechenden Komponenten softwaretechnisch umgesetzt. Das verwendete FTS *MiR100* besitzt ein eigenes WLAN Modul, über welches es mit der zugehörigen Transport Komponente kommuniziert. Diese wiederum ist auf einem externen PC umgesetzt. Die Koordinator - und Scheduler Komponente sowie alle Produkt Komponenten sind softwaretechnisch als jeweils einzelne Komponente auf einem Rechner implementiert. In Abbildung 67 sind die verwendeten Kommunikationsschnittstellen zwischen den physischen Fertigungsteilnehmern und den zugehörigen Komponenten dargestellt. Für die Kommunikation unter den einzelnen Komponenten der Steuerungsarchitektur wird das M2M-Protokolls *MQTT* Verwendet. Die genutzten Kommunikationsschnittstellen zwischen den Ressourcen -, Experten - sowie Transport Komponenten und den zugehörigen physischen Objekten sind fallabhängig. Sie hängen von den jeweiligen Kommunikationsschnittstellen ab, über welche die physischen Objekte angesteuert werden können. Die *MiR100* kommuniziert bspw. über *REST API* und der *Kuka iiwa* über *MQTT* mit der jeweiligen softwaretechnischen Komponente. Die Program-

miersprache, mit der die jeweilige Komponente implementiert wurde, ist unter den Komponenten aufgeführt. Auf die Beschreibung der einzelnen genannten Kommunikationsprotokolle sowie Programmiersprachen wird nicht genauer eingegangen, sondern auf entsprechende Fachliteratur verwiesen.

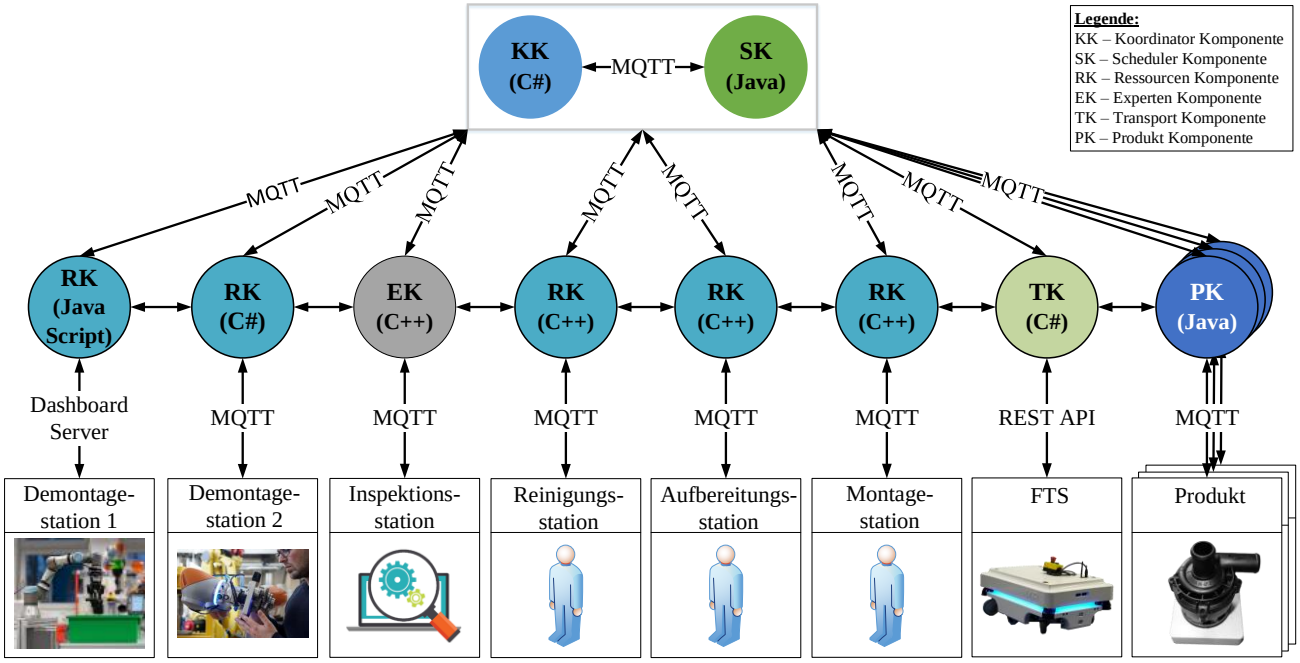


Abbildung 67: Vernetzung der Fertigungsteilnehmer sowie der Komponenten der hybriden Steuerungsarchitektur untereinander in der Modellfabrik. Die Komponenten kommunizieren ebenfalls über MQTT.

5.1.2 Versuchsszenario

Die exemplarische Refabrikation mehrerer Zusatzkühlwasserpumpen dient als Versuchsszenario zur Validierung der Steuerungsarchitektur innerhalb der Modellfabrik. Ziel ist es, dass die Steuerungsarchitektur die Modellfabrik so steuert, dass ein Auftragssatz, bestehend aus mehreren zu refabrizierenden Pumpen, diese erfolgreich durchlaufen kann. Die Steuerungsarchitektur darf währenddessen keine Randbedingungen innerhalb der einzelnen Aufträge verletzen. Zudem muss sie in der Lage sein auf unerwartet auftretende Ereignisse, in Form eines neuen, gültigen Ablaufplans, reagieren zu können. Die Zusatzkühlwasserpumpen finden in unterschiedlichen Fahrzeugen mehrerer Pkw-Hersteller Verwendung und bestehen aus zehn einzelnen Bauteilen. Diese sind folgend (mit Ausnahme der Schrauben) dargestellt:

- Torx-Schrauben T10 (viermal)
- Deckel (1)
- Rotor (2)
- Dichtung (3)
- Innenschale (4)
- Schale (5)
- Stator (6)

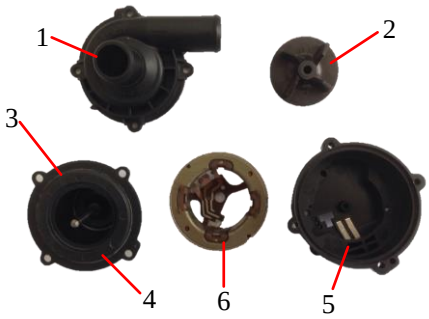


Abbildung 68: Bauteile der Zusatzkühlmittelpumpe.

Die Bauteile sind durch vier Schraubverbindungen und fünf Eingelegt-Verbindungen miteinander verbunden. Eine Schraubverbindung umfasst alle Bauteile, die durch eine Schraube miteinander verbunden sind sowie die Schraube selbst. Ein Bauteil das nur in eine Raumrichtung aus einem anderen Bauteil entnommen werden kann, ist durch eine Eingelegt-Verbindung mit diesem verbunden.

Zur exemplarischen Refabrikation muss die Zusatzkühlmittelpumpe mindestens die drei Stationen, Demontage, Inspektion und Montage, durchlaufen. In Abhängigkeit der Inspektionsergebnisse kann es notwendig sein, dass das Produkt zusätzlich die Reinigungsstation und/oder die Aufbereitungsstation durchlaufen muss. Hierfür werden die Pumpen unterschiedlich präpariert, um verschiedene Routen der Produkte durch die Modellfabrik zu generieren. Bezogen auf eine reale Refabrikation, repräsentiert dies unterschiedliche Defekte und Zustände der Produkte. Eine eigene Produkt Komponente wird für jedes, innerhalb der Modellfabrik befindliche, Produkt instanziiert. Jedes Produkt verfügt zur eindeutigen Identifikation über eine UID (Unique Identifier). Diese ist auf dem RFID-Chip der zugehörigen Transportbox hinterlegt. Die Produkte werden innerhalb der Modellfabrik in, dem Produkt fest zugeordneten, Boxen transportiert.

Die Prozesszeiten der einzelnen Operationen müssen der Steuerungsarchitektur zur Durchführung der SAMF bekannt sein. Die jeweilige Bearbeitung wird hierzu an jeder vorhandenen Station zehnmal durchgeführt. Der sich ergebende, durchschnittliche Wert wird für die Ablaufplanung genutzt und auf ganze Sekunden gerundet. Die Prozesszeiten für die exemplarische Refabrikation der Pumpe sind in Tabelle 15 aufgelistet. Die Prozesszeit der Lager Station wird mit 0 Sekunden angegeben, da hier lediglich eine neue Transportbox aufgeladen wird, und dieser Aufladevorgang bereits in der Transportzeit inkludiert ist. Die Reinigungs- und Aufbereitungsstation stellen exemplarische Stationen dar, weshalb die zugehörigen Prozesszeiten nicht auf realen Werten basieren, sondern angenommen werden. Der an diesen beiden Stationen befindliche Werker gibt jeweils ein Signal über die Prozessfertigstellung, sobald die entsprechende, angenommene Prozesszeit verstrichen ist. Für die Reinigung sind drei Prozesszeiten angegeben, wobei sich die erste Zeit auf die Reinigung des Deckels, die zweite des Rotors und dritte des Innengehäuses bezieht. Sind mehrere Bauteile zu reinigen, werden die entsprechenden Prozesszeiten addiert und die Summe der zugehörigen Operation zugeordnet.

Station	Durchschnittliche Prozesszeit [sec]
Lager	0
Demontage 1	120
Demontage 2	150
Inspektion	50
Reinigung	15 (Gehäuse), 20 (Rotor), 15 (Deckel)
Aufbereitung	60
Montage	70

Tabelle 15: Prozesszeiten der einzelnen Operationen zur exemplarischen Refabrikation der Zusatzkühlmittelpumpen in der Modellfabrik.

Drei Zusatzkühlwasserpumpen werden bei der Versuchsdurchführung initial in die Modellfabrik eingebracht. Die gewählte Anzahl an Produkten begründet sich dadurch, dass eine wesentlich höhere Anzahl die Kapazitäten der Modellfabrik überschreitet. Diese ist durch die Menge der Pufferplätze an den einzelnen Stationen restringiert. Der Prozessplan ist für alle Produkte anfangs identisch:

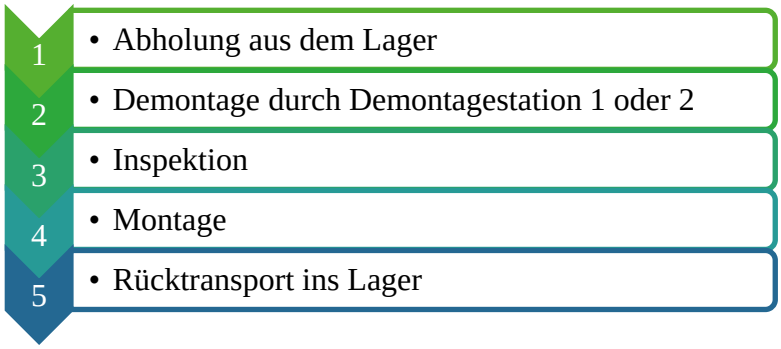


Abbildung 69: Initialer Prozessplan aller Produkt innerhalb der Modellfabrik

Erst während der Inspektion wird entschieden, welche weiteren Prozessschritte (Reinigung welcher Bauteile und/oder Aufbereitung) notwendig sind. Die Steuerungsarchitektur muss, basierend auf den Inspektionsergebnissen, die Prozesspläne der einzelnen Produkte und somit auch den gesamten Ablaufplan der Modellfabrik entsprechend anpassen. Die Produkte sind so präpariert, dass sich unterschiedliche, der Steuerungsarchitektur zuvor nicht bekannte, Inspektionsergebnisse und damit Prozesspläne ergeben. Vom Fertigungssteurer wird zudem nach einer gewissen Zeit ein neues Produkt instanziiert und in den laufenden Prozess eingeschleust, sodass sich ab diesem Zeitpunkt insgesamt vier Produkte im Umlauf befinden. Die Steuerungsarchitektur führt in beiden Fällen eine Neuplanung durch.

5.2 Versuchsdurchführung

Der Fertigungssteuerer startet bei der Versuchsdurchführung die Modellfabrik und initialisiert hierbei drei Zusatzkühlmittelpumpen. Die Steuerungsarchitektur führt daraufhin die SAMF aus und erzeugt den in Abbildung 70 dargestellten Ablaufplan. Alle drei Pumpen sind in diesem als Produkte 1, 2 und 3 eingeplant. Produkt 1 und 2 werden auf Demontagestation 1 und Produkt 3 auf Demontagestation 2 zerlegt. Die Farben der Blöcke im Ablaufplan entsprechen den Farben der Transportboxen des jeweiligen Produktes. Eine Überprüfung der Ablaufplanausführung ist dadurch visuell einfach möglich. Alle Pumpen befinden sich zu Beginn in der Lagerstation. Diese ist in Abbildung 70 nicht als Station aufgeführt, da dort kein Prozess ausgeführt wird. Nach dem Durchlaufen aller notwendigen Station werden die Pumpen final wieder an der Lagerstation abgeladen. Die Bearbeitung des zur jeweiligen Pumpe gehörenden Auftrags ist damit abgeschlossen.

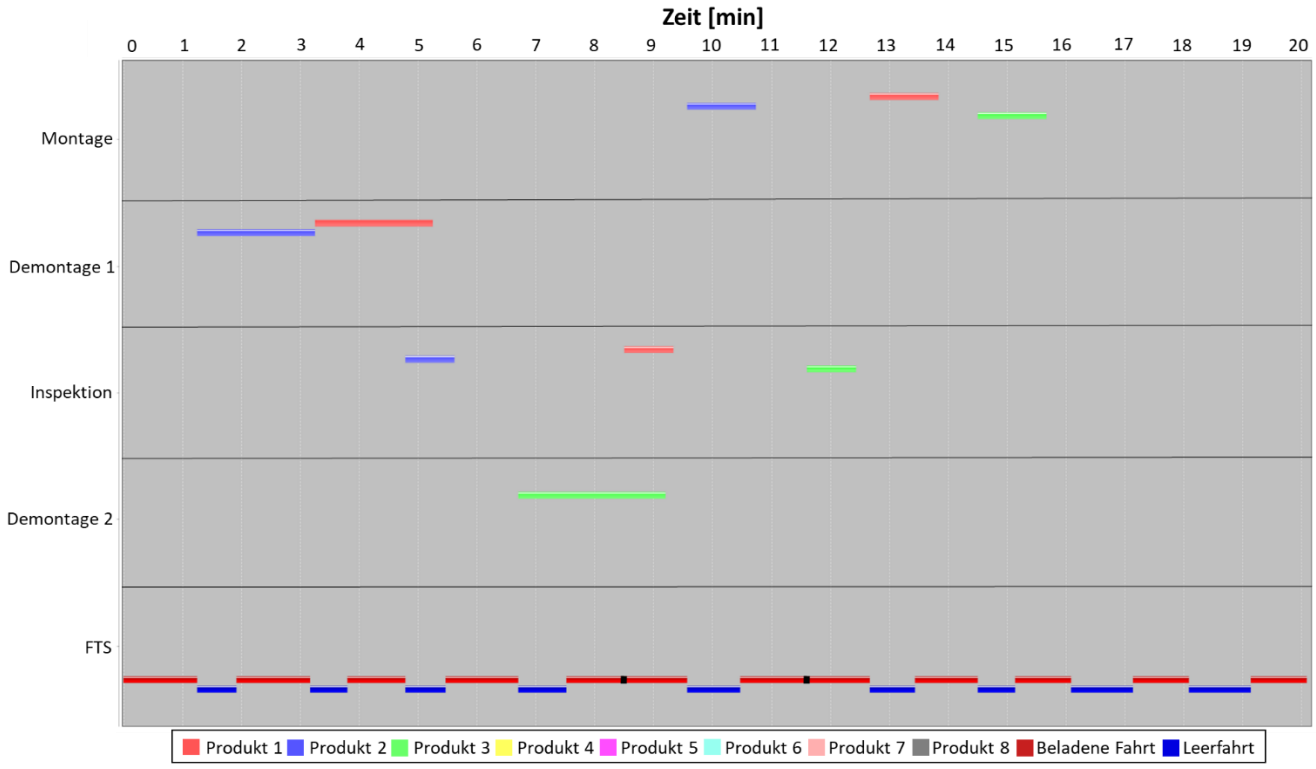


Abbildung 70: Initialer Ablaufplan zur Bearbeitung von drei Produkten innerhalb der Modellfabrik

Der erste durchzuführende Prozess innerhalb des Ablaufplans ist die Abholung von Produkt 1 aus dem Lager und der anschließende Transport diese zu Demontagestation 1. Das FTS befindet sich während des Startens der Modellfabrik bereits an der Lagerstation weshalb eine andernfalls notwendige Leerfahrt nicht im Ablaufplan enthalten ist. Die Transportfahrt von Produkt 2 aus dem Lager zur Demontagestation 1 entspricht dem ersten roten Block der Zeile *FTS* in Abbildung 70. Die Transport Komponente sendet den entsprechenden Transportbefehl an das FTS. Dieses setzt den Transport in seine Missionsliste und startet ihn anschließend. Eine Missionsliste mit der Mission *LadungAbladenS2P1* ist in Abbildung 71 zu sehen. Hierbei handelt es sich um einen Screenshot der Steuerungssoftware der MiR100. Die aufgeführte Mission Mission ist eine beladene Fahrt zum Abladen eines Produktes auf dem Pufferplatz 1 (P1 interne Bezeichnung) der Demontagestation 1 (interne Bezeichnung S2).

Abbildung 71: Warteliste der MiR100 zur Durchführung einer Transportfahrt.

Erfolgt nach 90-sekündiger Ausführung des initialen Ablaufplans die Anlegung eines neuen Produktes, das die gleichen Prozessschritte wie die übrigen drei Produkte durchlaufen soll, erstellt die hybride Steuerungsarchitektur den in Abbildung 72 zu sehenden Ablaufplan. Die drei Operationen des neu

hinzugekommenen Produktes 4 sind darin durch gelbe Blöcke dargestellt. Der neue Ablaufplan stellt die Demontage von Produkt 2 nicht mehr dar, da diese zum Zeitpunkt der Neuplanung in Bearbeitung befindlich ist und entsprechend nicht unterbrochen, sondern fertig bearbeitet wird. Von dem voraussichtlichen Fertigstellungszeitpunkt dieser Operation ist auch die Verfügbarkeit von Demontagestation 1 abhängig. Die zum Zeitpunkt der Neuplanung aktuelle Verfügbarkeit des FTS ist in dem neuen Ablaufplan ebenfalls berücksichtigt. Das FTS startet seine nächste Transportfahrt erst 61 Sekunden nach Beginn der Durchführung des neuen Ablaufplans. Dies entspricht der Zeit für die Leerfahrt zwischen der letzten bekannten Position des FTS (Demontagestation 1) und der Lagerstation (zur Aufnahme von Produkt 1).

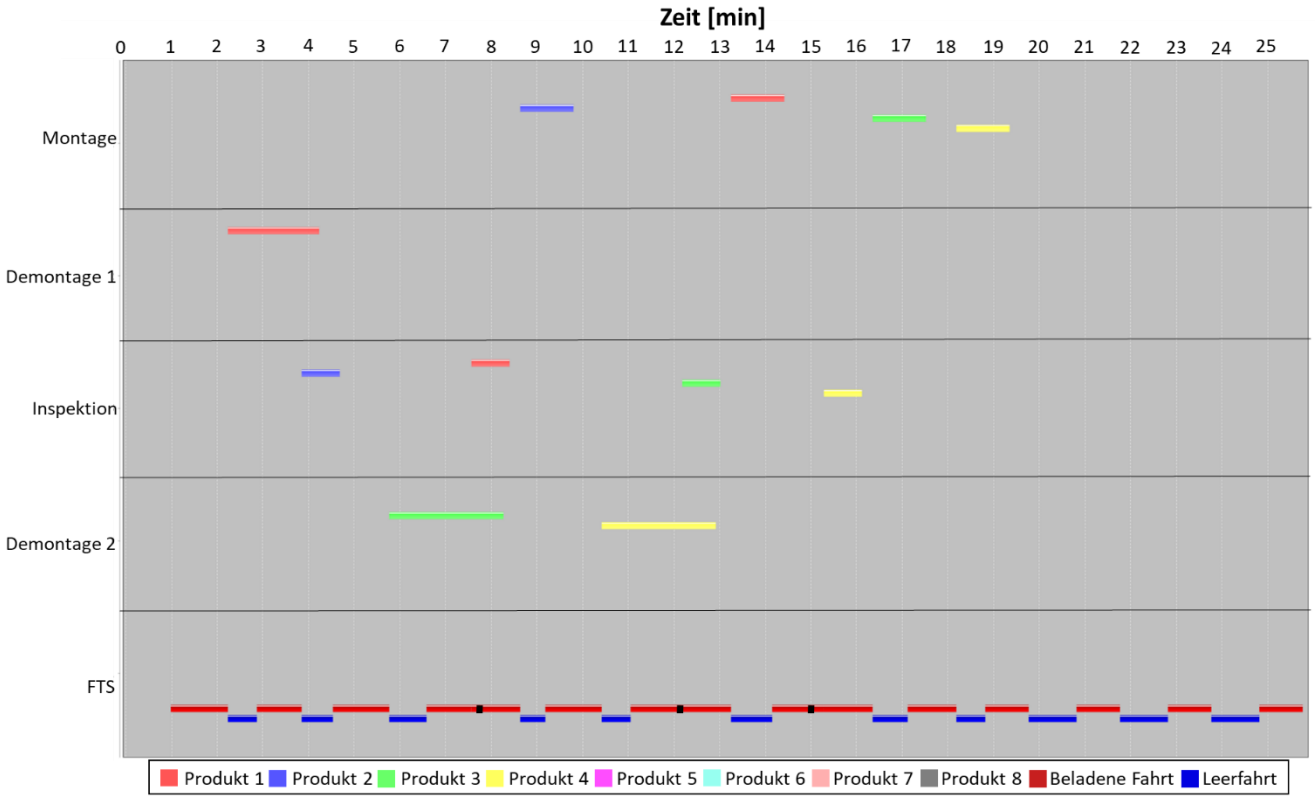


Abbildung 72: Neuer Ablaufplan inkl. des zusätzlich hinzugekommenen Produktes 4 (erste Leerfahrt zum Lager ist nicht grafisch dargestellt, aber wie zu sehen in der Verfügbarkeit des FTS berücksichtigt)

Produkte 2 gelangt nach weiterer Ausführung des neuen Ablaufplans zur Inspektionsstation. Ein Werker inspiziert dieses dort und entscheidet welche Teile gereinigt werden müssen und ob die Pumpe die Aufbereitungsstation durchlaufen muss. Der Werker kann seine Entscheidung der Experten Komponente über die in Abbildung 73 dargestellte GUI mitteilen. Die Experten Komponente erstellt und parametrisiert die zur Umsetzung dieser notwendigen Operationen und übermittelt sie der Produkt Komponenten des inspizierten Produktes. Abschließend teilt die Experten Komponente der Koordinator Komponente mit, dass für ein Produkt neue Operation notwendig sind. Diese initialisiert daraufhin eine Neuplanung.

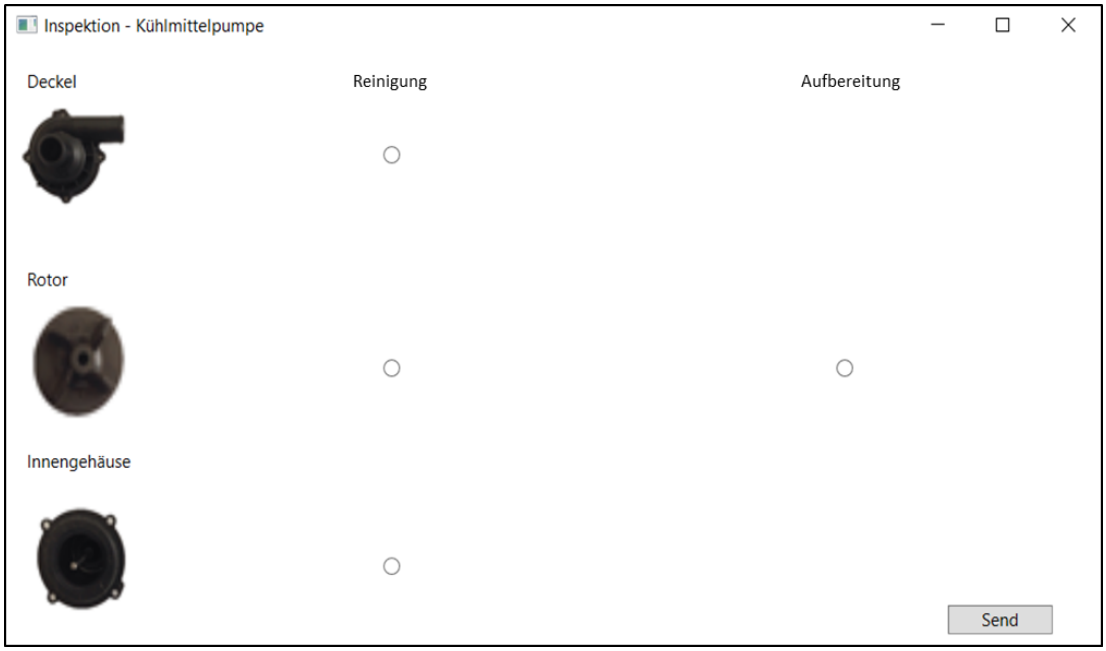


Abbildung 73: GUI zur Übermittlung der Inspektionsergebnisse an die Experten Komponente der Inspektionsstation.

Während der Versuchsdurchführung entscheidet der Werker, dass Deckel, Rotor und Innengehäuse von Produkt 2 gereinigt werden müssen. Der von der Steuerungsarchitektur daraufhin erzeugt Ablaufplan ist in Abbildung 74 dargestellt. Die zusätzliche, 50-sekündige Operation von Produkt 2 auf der Inspektionsstation ist darin entsprechend eingeplant. Es werden keine Randbedingung verletzt, sodass es sich um einen gültigen Ablaufplan handelt.

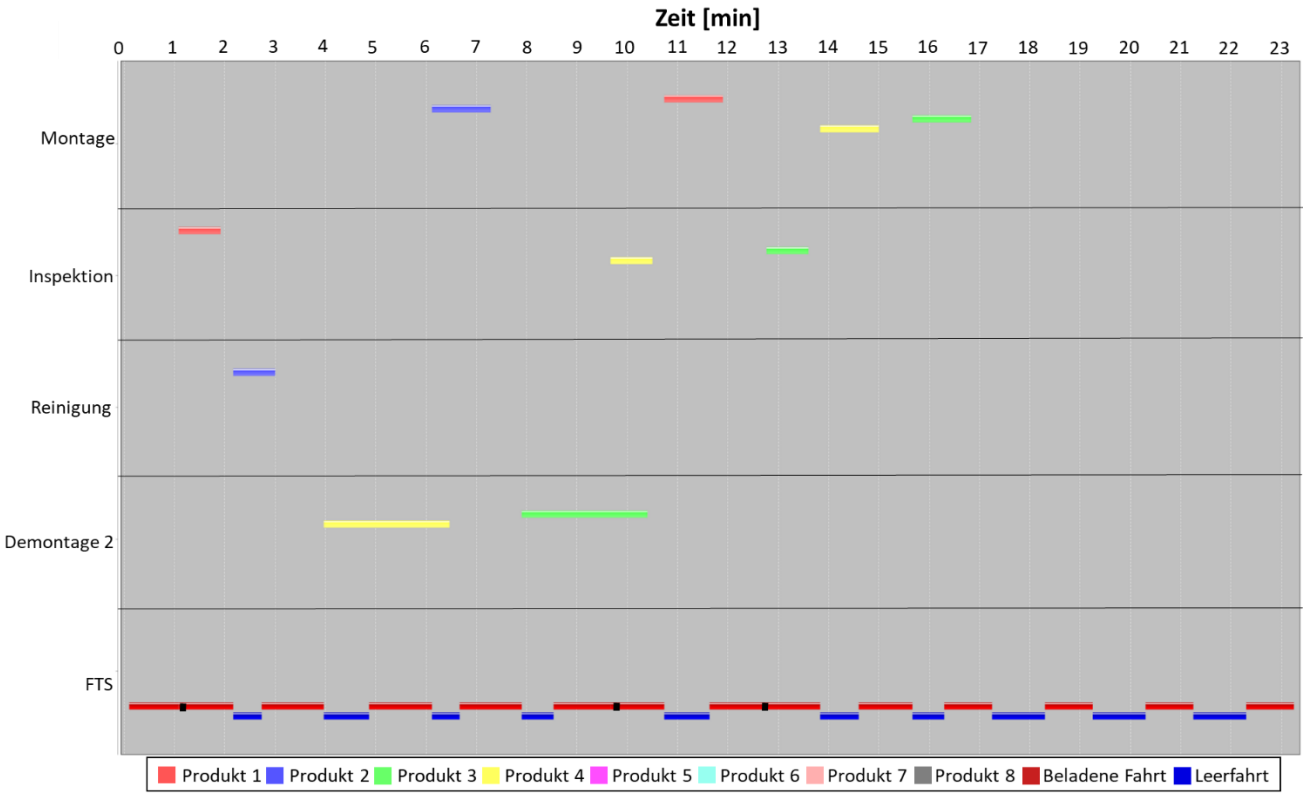


Abbildung 74: Neuer, gültiger Ablaufplan mit zusätzliche Reinigungsoperation für Produkt 2.

Die Steuerungsarchitektur kann auch weitere, unerwartet auftretende Ereignisse detektieren und auf diese reagieren. Sie kann bspw. nach einer Inspektion, neue notwendige Prozessschritte in eine Neuplanung inkludieren, oder auch den Ausfall einer der beiden Demontagestationen mitberücksichtigen.

Hierzu wird jeweils der Status sowie die Verfügbarkeit aller relevanten Fertigungsteilnehmer abgefragt. Die Steuerungsarchitektur erhält dadurch ein aktuelles, digitales Abbild der Modellfabrik. Basierend auf diesem Abbild wird, analog zu den vorangegangenen Simulationsstudien, eine Neuplanung durchgeführt.

5.3 Zwischenfazit

Die Integration der hybriden Steuerungsarchitektur in die Modellfabrik konnte erfolgreich umgesetzt werden. Hierfür wurden verschiedene Produkt – und Ressourcen Komponenten sowie jeweils eine Transport – und Experten Komponenten softwaretechnisch umgesetzt und mit ihren physischen Pendants vernetzt. Diese Vernetzung geschieht über fallabhängige Schnittstellen. Sie können über diese Schnittstellen sowohl Daten von den physischen Fertigungsteilnehmern extrahieren, als auch Steuerungsbefehle an diese weiterleiten. Zudem wurden die Koordinator – und Scheduler Komponente softwaretechnisch umgesetzt. Alle Komponenten können über das M2M Netzwerkprotokoll MQTT miteinander kommunizieren. Drei Zusatzkühlwasserpumpen sollten zur Validierung des Ansatzes die Modellfabrik durchlaufen, während diese von der hybriden Architektur gesteuert wird. Zusätzliche wurde nach einer gewissen Zeit eine neue, zu bearbeitende Zusatzkühlwasserpumpe in die Modellfabrik eingebracht. Diese dient zur Untersuchung der Reaktionsfähigkeit der Steuerungsarchitektur auf unerwartet auftretende Ereignisse. Die Steuerungsarchitektur muss zu diesem Zweck auch neue Inspektionsergebnisse, der im Umlauf befindlichen Produkt, adäquat im Rahmen einer Neuplanung berücksichtigen. Die Steuerungsarchitektur ist in der Lage alle zuvor aufgezeigten Aufgabenstellungen erfolgreich umzusetzen. Sie kann auf unerwartet auftretenden Ereignisse autonom in Form einer Neuplanung reagieren indem sie einen neuen, an die aktuelle Situation sowie den vorliegenden Bearbeitungsstatus aller Produkte angepassten, gültigen Ablaufplan erstellt. Ein Eingreifen durch den Fertigungssteuerer ist hierfür nicht notwendig.

Die erfolgreiche Implementierung der Steuerungsarchitektur innerhalb der Modellfabrik kann die Anwendbarkeit des vorgestellten Ansatzes in einem realitätsnahen RS validieren. Die prinzipielle Anwendbarkeit dieser in einem realen RS kann angenommen werden. Eine positive Beantwortung der 4. Forschungsfrage *„Ist die hybride Steuerungsarchitektur für den industriellen Einsatz prinzipiell geeignet ?“* ist damit möglich.

6 Zusammenfassung und Ausblick

In diesem Kapitel werden die in dieser Arbeit untersuchte Forschungsthematik sowie die erreichten Forschungsergebnisse abschließend zusammengefasst. Zusätzlich werden Impulse für mögliche weiterführende Forschungsaktivitäten gegeben.

6.1 Zusammenfassung und Diskussion der Forschungsergebnisse

Im Rahmen der vorliegenden Arbeit konnte die Lücke bzgl. einer speziellen Lösung zur Steuerung von RS mit flexiblem Materialtransport festgestellt werden. Die Notwendigkeit einer speziellen Lösung für diese Domäne ergibt sich durch den unbekannten Produktzustand der gebrauchten, zu refabrizierenden Produkte. Die produkt- und defektabhängigen Refabrikationsprozessschritte führen zu einem stark variierenden und vorab nicht bekannten Routing der Produkte durch das RS. Hinzu kommt der Trend zu variantenreichen und kundenindividuellen Produkten. Der Prozessplan zur Refabrikation eines Produktes muss, aufgrund des unbekannten Produktzustands, häufig während der Prozessausführung angepasst werden. Solche Änderungen müssen im Rahmen einer Neuplanung in den laufenden Betrieb integriert werden. Eine Neuplanung kann auch durch das Auftreten unerwarteter Ereignisse wie Maschinen- und FTS-Ausfälle, oder neue, zu bearbeitende Aufträge ausgelöst werden. Die Steuerungsarchitektur muss auf solche Ereignisse autonom reagieren können indem sie einen neuen, an die aktuelle Situation angepassten und gültigen Ablaufplans erstellt.

Aktuell verfügbare Steuerungssysteme können diesen Anforderungen nicht gerecht werden, was u. a. auf ihre zentrale Architektur und die damit verbundene geringe Flexibilität zurückzuführen ist. Zudem können zentrale Ansätze häufig nicht die Möglichkeiten ausschöpfen, die sich durch die Vernetzung der einzelnen Fertigungsteilnehmer im Kontext von Industrie 4.0 ergeben. Dezentrale Ansätze hingegen bieten eine höhere Flexibilität, haben allerdings den Nachteil, dass eine globale Optimierung des Systems nur schwer erreichbar ist. Diese Ansätze finden zudem oftmals keine Akzeptanz, da die vom System getroffenen Entscheidungen durch die selbststeuernde Charakteristik vom Fertigungssteuerer oftmals nicht nachzuvollziehen sind.

Ein flexibler Materialtransport, bspw. durch FTS, ist notwendig um die individuellen Routen der Produkte durch das RS zu realisieren. Die Einbindung der FTS-Steuerung in das bestehende Produktionssteuerungssystem erfolgt meist durch Flottenmanagement-Systeme. Diese führen die Ablaufplanung der notwendigen Transporte, basierend auf einem vom Produktionssteuerungssystem erhalten Maschinenbelegungsplan, durch. Beide Ablaufplanungsprobleme werden, obwohl sie sich gegenseitig beeinflussen und voneinander abhängig sind, sequenziell durchgeführt. Verfügbare Produktionssteuerungssystem bieten keine Möglichkeit diese Ablaufplanungsprobleme simultan zu betrachten, was Optimierungspotenzial offenlässt. Auch in der wissenschaftlichen Fachliteratur ist die SAMF nur wenig erforscht. In den Veröffentlichungen zu diesem Thema erfolgt die Entwicklung der Optimierungsalgorithmen fast ausschließlich für deterministische Umgebungen. Die Reaktionsfähigkeit der Ansätze auf unerwartet eintretende Ereignisse ist meist nicht gegeben, für die praktische Anwendbarkeit allerdings von essentieller Bedeutung.

Um die notwendige Flexibilität des Steuerungssystems zu erreichen und dennoch die Probleme einer dezentralen Steuerungsarchitektur zu vermeiden, wurde in dieser Arbeit eine hybride Steuerungsarchitektur vorgestellt. Sie besteht aus verschiedene Komponenten und Komponententypen.

Die Koordinator Komponente sammelt Informationen über den aktuellen Status sowie die Verfügbarkeit aller Fertigungsteilnehmer und hat dadurch einen Gesamtüberblick über das RS. Sie kann durch ihren Gesamtüberblick unerwartet auftretende Ereignisse erkennen und entsprechende Prozess zur Problemlösung initiieren. Ihre Aufgabe besteht in der Koordinierung der weiteren Komponenten, wenn bspw. eine Neuplanung durchgeführt werden muss. Die Scheduler Komponente erstellt und optimiert die SAMF. Diese beiden Komponenten stellen die zentrale Ebene der Architektur dar.

Die verschiedenen Komponenten der dezentralen Ebene repräsentieren bzw. steuern die einzelnen Fertigungsteilnehmer wie bspw. Maschinen, FTS und sich im Umlauf befindliche Produkte. Die Fertigungsteilnehmer sind hierzu über geeignete, fallabhängige Schnittstellen mit den zugehörigen softwaretechnischen Komponenten verbunden. Diese können über eine M2M-Schnittstelle miteinander kommunizieren, wodurch sie ein CPS bilden. Alle Steuerungsaufgaben, welchen keinen zwingenden Gesamtüberblick des RS benötigen, finden auf der dezentralen Ebene statt. Hierzu gehören bspw. die Demontageprozessplanung, die FTS-Pfadplanung sowie die Identifikation des an einer Ressource zur Bearbeitung vorliegenden Produktes. Die Einführung der Experten Komponenten ermöglicht es, zusätzlich durchzuführende Operationen für ein Produkt während der Prozessausführung in die Ablaufplanung zu integrieren. Dies kann notwendig sein, wenn im Rahmen eines Inspektionsprozesses festgestellt wird, dass zur Refabrikation eines Produktes zusätzliche Prozessschritte notwendig sind. Die dezentrale Ebene kann beim Ausfall der zentralen Ebene, oder während sich diese in einer Neuplanung befindet, den aktuellen Ablaufplan bis zu dessen Beendigung autonom weiter ausführen.

Ein CP-Modell zur SAMF wurde zielführend entwickelt und in der Scheduler Komponente erfolgreich implementiert. Die anhand von BI durchgeführten Simulationsstudien zeigen, dass die simultane, gegenüber der sequenziellen, Ablaufplanung eine DLZ-Reduzierung von 35,6 % ergeben kann. Gegenüber Ansätzen der reinen Selbstorganisation konnte sogar eine durchschnittliche DLZ-Reduzierung in Höhe von 40,3 % nachgewiesen werden. Im Vergleich zu anderen veröffentlichten Ansätzen zur SAMF zeigt sich, dass das entwickelte CP-Modell mindestens gleichgute, oder meist bessere Lösungen in einer deutlich geringeren RZ liefert. Neben der DLZ-Minimierung wurden auch andere Zielfunktionen untersucht und implementiert. Hierzu gehören die durchschnittliche MA, sowie multikriterielle, lexikographische Zielfunktionen die MA und DLZ auf unterschiedliche Arten kombinieren. Die lexikografische Zielfunktion mit der DLZ als primäres und der MA als sekundäres Ziel liefert das beste Ergebnis für den Kompromiss zwischen DLZ und MA.

Der vorgestellte Ansatz ist, im Gegensatz zu den meisten Veröffentlichungen zur SAMF, in der Lage auf unerwartet eintretende Ereignisse zu reagieren. Er aggregiert hierzu den Status sowie die Verfügbarkeit aller relevanten Fertigungsteilnehmer, um ein aktuelles, digitales Abbild des RS zu erhalten und führt, basierend auf diesen Informationen eine Neuplanung durch. Zur Verifikation der Reaktionsfähigkeit der Steuerungsarchitektur wurden diverse Simulationsstudien durchgeführt. Folgende, unerwartet auftretende Ereignisse wurden untersucht:

- Maschinenausfall
- FTS-Ausfall
- Neuer, zu bearbeitender Auftrag
- Neue, zusätzliche und in den Ablaufplan zu integrierende Operation für ein Produkt

Diese Ereignisse wurden sowohl als einzelne, unabhängig voneinander, als auch als nacheinander eintretende Ereignisse simuliert. Die neu entwickelte Steuerungsarchitektur ist in der Lage, auf alle diese

Ereignisse adäquat, in Form eines neuen, angepassten und gültigen Ablaufplans zu reagieren. Die Durchführung der Neuplanung erfolgte innerhalb nur einer Sekunde. Zudem konnte gezeigt werden, dass während der Prozessausführung neu hinzukommende Aufträge in den aktuellen Ablaufplan integriert werden. Anderen Ansätzen hängen neue Aufträge erst nach Abarbeitung der aktuell im Umlauf befindlichen Aufträge dem Ablaufplan an.

Die erfolgreiche Implementierung der hybriden Steuerungsarchitektur in einer hierzu errichteten Modellfabrik konnte ebenfalls demonstriert werden. Ziel ist die Steuerung einer exemplarischen Refabrikation von Zusatzkühlwasserpumpen. Alle Komponenten der Architektur wurden softwaretechnisch umgesetzt und mit den einzelnen Fertigungsteilnehmern verbunden. Die Kommunikation zwischen den einzelnen Komponenten erfolgt über das standardisierte M2M-Kommunikationsprotokolle MQTT. Alle Zusatzkühlmittelpumpen können die Modellfabrik komplett durchlaufen. Beim Auftreten unerwarteter Ereignissen ist die Steuerungsarchitektur in der Lage, autonom einen neuen, gültigen Ablaufplan zu erstellen und auszuführen. Die prinzipielle Anwendbarkeit dieser in einem realen RS kann durch die Validierung der Architektur in der Modellfabrik angenommen werden. Die aufgezeigten Vorteile des entwickelten Steuerungsansatzes sind abschließend aufgelistet:

Nummer	Vorteil
1.	Deutliche DLZ-Reduzierung (35,6 %) durch simultane gegenüber sequentieller Ablaufplanung von Maschinen und FTS
2.	CP-basiertes Optimierungsverfahren zur SAMF liefert gleich gute, oder bessere Ergebnisse als andere SotA-Verfahren bzgl. DLZ-Minimierung und hierfür notwendiger RZ
3.	Adäquate Reaktion auf unerwartet auftretende Ereignisse innerhalb des RS unter Nutzung von Echtzeit-Daten möglich
4.	Aufrechterhaltung der Prozessausführung bei kurzfristigem Ausfall der zentralen Ebene, oder während einer Neuplanung durch dezentrale Steuerungskomponenten
5.	Prinzipielle, industrielle Anwendbarkeit durch Steuerung einer Modellfabrik validiert

Tabelle 16: Vorteile der hybriden Steuerungsarchitektur

Die Steuerungsarchitektur eignet sich für die Anwendung in RS, die sich durch folgende Merkmale charakterisieren lassen:

- Hohe Produktvarianz und/oder –individualisierung bis hin zu Losgröße 1
- Flexibler Materialtransport durch FTS
- Taktzeiten im Bereich von einigen Sekunden bis Minuten
- Matrixstrukturelles Fertigungslayout

Mit den hier zusammengefassten Ergebnissen der vorliegenden Arbeit ist es möglich, die in Kapitel 2.5 aufgestellten Forschungsfragen positiv beantworten zu können und somit auch die aufgestellte Forschungshypothese (*Die Verwendung einer hybriden, dezentral-hierarchischen Steuerungsarchitektur, mit der Integration simultaner Ablaufplanung von Maschinen und FTS, erlaubt es, die Vorteile einer zentralen und dezentralen Produktionssteuerung zu kombinieren, sodass sowohl eine globale*

Optimierung erreicht werden kann, als auch die vorliegende Stochastik und Dynamik innerhalb eines RS bewältigt werden kann.) zu bestätigen.

Im folgenden Kapitel wird ein Ausblick über denkbare Weiterentwicklungen der vorgestellten Arbeiten gegeben.

6.2 Ausblick

Die folgenden drei Weiterentwicklungsrichtungen stellen vielversprechende Ansätze zur Verbesserung der Qualität und Anwendbarkeit der hybriden Steuerungsarchitektur dar:

- Erweiterungen des Ansatzes um eine lernende Komponente unter Nutzung von Methoden der KI und des maschinellen Lernens.
- Weiterentwicklung des Ansatzes zu einem kommerziellen Feinplanungs- und Steuerungsmodul
- Implementierung und Test des Feinplanungsmoduls in einer industriellen (Re-)fabrikation.

Die letzten beiden aufgezeigten Richtungen beschäftigen sich mit der Erweiterung der Nutzbarkeit der entwickelten Steuerungsarchitektur im industriellen Einsatz. Der erste Punkt zielt hingegen auf eine generelle Weiterentwicklung des vorgestellten Ansatzes ab. Folgend werden die einzelnen Richtungen beschrieben sowie deren Zusammenhang erläutert.

Jede Ablaufplanung ist maximal so gut wie die Daten, auf denen Sie basiert. Das bedeutet, dass ein Ablaufplan nur so gut ist, wie die Genauigkeit der angenommenen Bearbeitungs- und Fahrtzeiten im Vergleich zu den späteren tatsächlichen Werten es erlaubt. Stark schwankende Prozesszeiten sind in der Refabrikation ein häufig auftretendes Problem. Diese Schwankungen begründen sich durch die variierenden Zustände der zu refabrizierenden Produkte. Um trotz dieser Schwankungen möglichst präzise Aussagen über die Prozesszeiten treffen zu können, kann der Einsatz maschinellen Lernens als vielversprechender Ansatz dienen. Über Methoden des maschinellen Lernens kann im einfachsten Fall, für ein bestimmtes Produkt oder Bauteil, die Prozesszeit bestimmt werden, die mit der höchsten Wahrscheinlichkeit zu erwarten ist. Weitere Indikatoren zur Prognose der Prozesszeit sind bspw.:

- Nutzungsdauer
- Alter des Produktes
- Einsatzort des Produktes
- Bekannte Schwachstellen
- Einsatzgebiet des Produktes
- Ergebnisse einer Eingangsinspektion
- Evtl. Sensormesswerte, die während der Nutzung des Produktes aufgezeichnet wurden

Die Nutzung dieser Indikatoren kann zu einer genaueren Prognose der wahrscheinlichen Prozesszeit führen. Eine Prognose der Fahrzeiten kann über einfache statistische Methoden, wie die Medianbestimmung, erfolgen. Grund hierfür ist die geringe Schwankung der Fahrzeiten. Größere Schwankungen ergeben sich nur durch unvorhergesehen eintretende Ereignisse wie bspw. plötzlich auf dem geplanten Pfad befindliche Objekte. Diese Ereignisse unterliegen meist keinem genauen Muster, sodass Methoden des maschinellen Lernens diesbezüglich nur bedingt Vorhersagen ermitteln können. Durch die Nutzung des Medians werden solche Ausreißer nicht berücksichtigt.

Die Steuerungsarchitektur kann zur besseren Anwendbarkeit im industriellen Umfeld, in ein kommerzielles Feinplanungs- und Steuerungsmodul weiterentwickelt werden. Da, nach bestem Wissen des Autors, in den aktuell verfügbaren Produktionssteuerungssystemen die Ablaufplanung von Maschinen und FTS nur sequenziell, nicht aber simultan durchgeführt wird, könnte ein solches Modul für den Kunden den Vorteil der DLZ-Reduktion bedeuten. Des Weiteren kann das Modul der autonomen Reaktion auf unerwartet auftretenden Ereignisse dienen und diese durch entsprechende Steuerungseingriffe lösen. Hierzu ist die Kollaboration mit einem Produktionssteuerungs- oder Softwareanbieter ratsam. Dieser verfügt über die notwendige Expertise in der kommerziellen Softwareentwicklung und kann zudem die Bereiche Vertrieb, Wartung sowie Support unterstützen bzw. übernehmen. Eine Forschungseinrichtung kann diese Bereiche nur bedingt und oftmals nicht übernehmen.

Die Implementierung sowie das Testen der Steuerungsarchitektur in einem realen RS oder einem Teilbereich dieses, wird ebenfalls als wichtiger zukünftiger Schritt bewertet. Durch die erfolgreiche Implementierung des Ansatzes zur Steuerung der Modellfabrik konnte der Einsatz in einer realitätsnahen Umgebung bereits bewiesen werden. Die Produktionssteuerung ist allerdings ein sehr sensibler Bereich, da ein Ausfall oder eine Fehlfunktion dieser zu kostenintensiven Stillständen führen kann. Aus diesem Grund wird empfohlen, die Steuerungsarchitektur im ersten Schritt in einem Teilbereich eines RS zu integrieren. Die zuvor beschriebene Weiterentwicklung des Ansatzes zu einem kommerziellen Feinplanungsmodul sollte idealerweise vorab erfolgen. Dadurch können evtl. Ausfälle oder Fehlfunktionen und die damit einhergehenden Fertigungsstillstände reduziert werden. Die Integration solcher Systeme in eine bestehende Fertigung ist in der Regel zeitintensiv, weshalb empfohlen wird, diesen Schritt in einem größer angelegten Projekt durchzuführen. Die Einführung eines neuen MES nimmt bspw. bis zu einem Jahr in Anspruch [188]. Nachdem der Ansatz in einem industriellen RS erfolgreich integriert und validiert wurde, könnte das entstandene Feinplanungs- und Steuerungsmodul einer kommerziellen Vertreibung unterzogen werden.

Eine kontinuierliche Weiterentwicklung der präsentierten Modellfabrik ist ebenfalls geplant. Neben der Umlegung dieser in einen größeren Showroom, soll sie mit zusätzlichem Equipment sowie im Rahmen anderer Projekte entwickelten Demonstratoren erweitert werden. Die Modellfabrik dient zum einen der Außendarstellung geleisteter Forschungsbeiträge bei Besuchern aus Industrie und Forschung. Zum anderen kann sie für Schulungszwecke von Studierenden und Interessierten aus der Industrie in den Bereichen Robotik und Industrie 4.0 genutzt werden. Die vorgestellte hybride Steuerungsarchitektur wird weiterhin zur Steuerung der Modellfabrik eingesetzt. Hierdurch können alle Neuerungen an der Steuerungsarchitektur direkt innerhalb der Modellfabrik getestet und validiert werden.

Abschließend lässt sich zusammenfassen, dass im Rahmen dieser Arbeit wichtige und zielführende Schritte zur Realisierung einer Steuerungsarchitektur für die variantenreiche und kundenindividuelle (Re-)fabrikation erfolgreich entwickelt, umgesetzt und validiert wurden. Diese bilden das Fundament zur effizienten und zielgerichteten Beherrschung der steigenden Flexibilitätsansprüche in der (Re-)fabrikation.

7 Anhang

Anhang A - Liste wissenschaftlicher Veröffentlichungen:

[189] S. Groß, W. Gerke, and P. Plapper, “Human-Robot-Collaboration for dismantling processes,” in *Robotix-Academy Conference for Industrial Robotics (RACIR)*, 2017, pp. 9–12.

[190] S. Groß, W. Gerke, und P. Plapper, “Mensch-Roboter-Kollaboration für Demontageprozesse,” in *AALE 2018*, 2018th ed., Köln: VDE-Verlag, 2018.

[191] S. Groß, W. Gerke, and P. Plapper, “A survey: Scheduling of Automated Guided Vehicles in Flexible (Re-)Manufacturing Systems,” in *Robotix-Academy Conference for Industrial Robotics (RACIR) 2019*, 2019.

[192] S. Groß, W. Gerke, and P. Plapper, “Simulation-based optimization using multi-agent technology for efficient and flexible production planning and control in remanufacturing,” in *International Conference on Remanufacturing 2019*, 2019.

[193] S. Groß, W. Gerke, and P. Plapper, “Optimized and flexible scheduling of AGVs and process machines in Remanufacturing 4.0 Systems using multi-agent technology and simultaneous scheduling,” in *Abstracts III International Workshop on Autonomous Remanufacturing*, F. J. Ramírez Fernández and A. Honrubia Escribano, Eds. 2019, p. 23.

[194] S. Groß, W. Gerke, and P. Plapper, “Agentenbasierte, hybride Steuerungsarchitektur für cyberphysische Refabrikationssysteme,” in *Tagungsband AALE 2020*, 2020.

[195] S. Groß, T. Bartscherer, A. Pereira, W. Gerke, and P. Plapper, “Mensch-Roboter-Kollaboration in der Domäne Refabrikation – State-of-the-Art und Ausblick,” in *Tagungsband AALE 2020*, 2020.

[196] S. Groß, W. Gerke, and P. Plapper, “Agent-based, hybrid control architecture for optimized and flexible production scheduling and control in remanufacturing”, in *Journal of Remanufacturing*, vol. 27, no. 1, Springer, 2020, DOI: 10.1007/s13243-020-00081-z

Anhang B - Beteiligung an Forschungsanträgen:

- Robots for Industrial Assistance Tasks (ROBIAT); Call: H2020-ICT-2016-2017 → nicht für eine Förderung ausgewählt
- Automatische, robotergeführte optische Inspektion geflochtener und lasergeschnittener Stents (InStent); Forschung an Fachhochschulen – FHprofUnt → genehmigt, offizieller Projektstart 01.10.2019, Projektende: 30.09.2022, **Zuwendung: 418.727,22€**
- Adaptive Robots in Flexible Hybrid Production (ARoP); Innovative Training Networks (ITN); Call: H2020-MSCA-ITN-2019 → nicht für eine Förderung ausgewählt, Neueinreichung geplant
- Enhancing the technology of remanufacturing, refurbishment, reuse and in-situ repair, with intelligent maintenance to increase the return on investment for large-scale equipment (ENTRE); Call ID: DT-FOF-06-2019: Refurbishment and re-manufacturing of large industrial equipment (IA); H2020 → nicht für eine Förderung ausgewählt

- Innovations towards effective solutions for remanufacturing, reuse and recycling of electric vehicle battery packs (LIFE REBAT); Call: LIFE Environment and Resource Efficiency project application; LIFE 2019 → nicht für eine Förderung ausgewählt
- Robotix-Academy Verlängerungsantrag; EFRE; Interreg VA Großregion → genehmigt, Projektverlängerung bis 30.06.2022, **Zuwendung: 342.240,00€**
- Human-Machine Intelligence for Timely and Efficient (Re-)manufacturing (MIREMA) Topic: Artificial intelligence for manufacturing; Call: Information and Communication Technologies; Call ID: H2020-ICT-2018-20 → nicht für eine Förderung ausgewählt
- Virtual and Physical Human-Robot-Cooperation (VIP-HRC); German-French joint call for proposals on "Artificial Intelligence" → in Bewertung befindlich

Anhang C - Anderweitige Projektakquise:

- START Wettbewerb 2018 - EU-Forschung an der Hochschule Trier, 1. Platz, **15.000€**
- START Wettbewerb 2019 - EU-Forschung an der Hochschule Trier, 1. Platz, **15.000€**
- K.-H. Müller Präzisionswerkzeuge GmbH

Anhang D - Betreute studentische Arbeiten:

- Alexander Ermisch und Wladislaw Sontag: *„Erstellung eines Praktikumsversuchs für den Universal Robots UR3 für die Lehrveranstaltung Robotik mit Praktikum"*, Projektarbeit im Bachelorstudium, 02/2017 – 05/2017
- Jochen Mees: *„Entwicklung, Aufbau und Inbetriebnahme einer Roboterzelle zur Mensch-Roboter-Kollaboration"*, Praktische Studienphase, 07/2017 – 09/2017
- Abellah Idlasri: *„Entwicklung einer MRK-Applikation zum Entlöten von Bausteinen aus Leiterplatten im Bereich der Leistungselektronik"*, Praktische Studienphase und Bachelorthesis, 09/2018 – 02/2019
- Christopher Huth: *„Planung und Simulation eines Remanufacturing-Prozesses"*, Projektarbeit im Masterstudium, 07/2019 – 09/2019
- Lukas Vogt: *„Aufbau und Inbetriebnahme einer Modellfabrik zur Refabrikation"*, Praktische Studienphase, 09/2019 – 12/2019
- Sven Holwein und Johannes Richters: *„Bauteil Erkennung und Entnahme von einer mobilen Plattform mittels Industrieroboter und Vision System"*, Projektarbeit im Bachelorstudium, 10/2019 – 12/2019
- Rachid el Harrar: *„Konzeptionierung und Umsetzung einer Neuplanung des Robotik Technikum am Umwelt-Campus Birkenfeld der Hochschule Trier",* Praktische Studienphase, 11/2019 – 02/2020
- Nicolas Kiebel und Martin Seidinger: *„Entwicklung und Implementierung einer elektrischen Hubeinheit auf die MiR 100 inkl. der Entwicklung einer Strategie zum Aufnehmen und Absetzen von Transportboxen sowie Einbindung dieser in die Agentenbasierte Steuerungssoftware der Modellfabrik"*, Projektarbeit im Masterstudium im Rahmen der Vorlesung *Übungen zur Robotik und Mechatronik*, 11/2019 – 02/2020
- Danja Steinberg und Julien Murach: *„Demontage einer Kühlmittelpumpe unter Verwendung einer Multi-Agenten basierten Steuerungssoftware für den KUKA LBR iiwa und Einbindung des Demonstrators in die Modellfabrik als zusätzliche Demontagestation"*, Projektarbeit im

Masterstudium im Rahmen der Vorlesung *Übungen zur Robotik und Mechatronik*, 11/2019 – 02/2020

- Kevin Wagner und Lukas Vogt: *„Entwicklung einer Vision basierten Anwesenheitskontrolle mit Einbindung in die Modellfabrik über MQTT“,* Projektarbeit im Masterstudium im Rahmen der Vorlesung *Übungen zur Robotik und Mechatronik*, 11/2019 – 02/2020
- Lukas Vogt: „*Aufbau und Inbetriebnahme einer Modellfabrik zur Refabrikation*“, Bachelorthesis, 01/2020 – 02/2020
- Johannes Richters: *„Aufbau und Inbetriebnahme von jeweils zwei Pufferplätzen an den drei Roboterstationen der Modellfabrik zur Refabrikation am Umwelt-Campus Birkenfeld“*, Praxisorientiertes Arbeiten im Bachelorstudium 04/2020 – 05/2020
- Levin Czenkusch: *„Vision basiertes Greifen einer Kühlmittelpumpe mit dem KUKA LBR iiwa“*, Projektarbeit im Masterstudium im Rahmen der Vorlesung *Übungen zur Robotik und Mechatronik*, 04/2019 – 06/2020
- Steven Hohmann: *„Entwicklung eines Roboter-Tools zum Greifen von Transportboxen sowie Durchführung einer Erreichbarkeitsanalyse“*, Projektarbeit im Masterstudium im Rahmen der Vorlesung *Übungen zur Robotik und Mechatronik*, 04/2019 – 06/2020
- Johannes Richters: *„Entwicklung, Beschaffung, Aufbau und Inbetriebnahme einer Lagerstation für die Modellfabrik zur Refabrikation am Umwelt-Campus Birkenfeld“*, Fachprojekt im Bachelorstudium 05/2020 – 07/2020
- Florian Schäfer: *„Integration des intelligenten Demontageassistenten Agenten (IDAA) in die agentenbasierte Steuerungsarchitektur der Modellfabrik“*, Masterthesis, 05/2020 – 10/2020
- Leon Grohs: *„Entwicklung eines Refabrikationskonzeptes für E-Scooter“,* Bachelorthesis, 09/2020 – 01/2021
- Lukas Vogt: *„Entwicklung dezentraler Smart Devices am Beispiel Beamer zur Werkerführung“*, Interdisziplinäre Projektarbeit im Masterstudium, 09/2020 – 11/2020
- Kristof Ueding und Bastian Urschel: *„Entwicklung einer Robotic as a Service (RaaS) Anwendung unter Nutzung von Distributed-Ledger-Technologie“*, Projektarbeit im Masterstudium im Rahmen der Vorlesung *Übungen zur Robotik und Mechatronik*, 10/2020 – 02/2021
- Arian Schneberger und Jonathan Schneider: *„Entwicklung einer Applikation zur Erkennung der Werkerbeanspruchung mittels Smart Watch und Gestenerkennung“*, Projektarbeit im Masterstudium im Rahmen der Vorlesung *Übungen zur Robotik und Mechatronik*, 10/2020 – 02/2021
- Kevin Frank Gnädig und Swen Haab: *„Entwicklung eines smarten Robotergreifers“,* Projektarbeit im Masterstudium im Rahmen der Vorlesung *Übungen zur Robotik und Mechatronik*, 10/2020 – 02/2021
- Julian Räsch, Luca Zimmer, Florian Schorr und Simon Högner: *„Entwicklung eines Konzeptes zur automatisierten Demontage von Hochvolt-Batterien aus E-Fahrzeugen“,* Fachprojekt im Bachelorstudium 10/2020 – 01/2021

Anhang E - Betreuung von Lehrveranstaltungen:

- Durchführung von Praktikumsversuchen für die Bachelorvorlesung „Robotik mit Praktikum“, WS16/17
- Durchführung von Praktikumsversuchen für die Bachelorvorlesung „Elektrische Maschinen mit Praktikum“, WS16/17

- Durchführung von Praktikumsversuchen für die Bachelorvorlesung „Robotik mit Praktikum“, WS17/18
- Durchführung von Praktikumsversuchen für die Bachelorvorlesung „Elektrische Maschinen mit Praktikum“, WS17/18
- Übungen für die Bachelorvorlesung „Robotik mit Praktikum“, WS18/19
- Vorlesung und Übungen für die Mastervorlesung „Übungen zur Robotik und Mechatronik“, WS18/19
- Ausgewählte Übungen für die Mastervorlesung „Prozessleittechnik“, SS19
- Übungen für die Mastervorlesung „Robotik und virtuelle Planung“, SS19
- Vorlesung und Übungen für die Mastervorlesung „Übungen zur Robotik und Mechatronik“, WS19/20
- Übungen in der Veranstaltung „Lab Works – Robotics“, SS20
- Vorlesung und Übungen für die Mastervorlesung „Übungen zur Robotik und Mechatronik“, SS20
- Vorlesung und Übungen für die Mastervorlesung „Übungen zur Robotik und Mechatronik“, WS20/21

Anhang F - Ökologische und ökonomische Vorteile der Refabrikation:

Aufgrund von begrenzten Ressourcen und der gesetzlichen Verpflichtung von Großunternehmen zur Vorlegung einer Nachhaltigkeitsstrategie, erlangt das Recycling und vor allem die Refabrikation, auch Remanufacturing genannt, von Produkten eine steigende Bedeutung. Beim Recycling wird, im Gegensatz zur Refabrikation, nur eine Wiederverwendung der Rohstoffe innerhalb eines Produktes angestrebt. Die Form und Funktion des Produktes geht folglich verloren. Ein Beispiel für eine solche gesetzliche Verpflichtung ist das von der EU erlassene Gesetz zur „Offenlegung nicht finanzieller Informationen“. Dieses betrifft Firmen ab einer Größe von 500 Mitarbeitern und verpflichtet diese u. a. zur Berichterstattung über Strategie, Risiko und Ergebnisse im Bereich Umwelt. Das Gesetz wurde am 1. Januar 2017 gültig [197]. Durch das Recycling werden zum einen Rohstoffe und zum anderen Energieaufwände eingespart. Diese sind notwendig, um Rohstoffe zu gewinnen, zu transportieren und evtl. erforderliche Veredlungsprozesse durchzuführen. Dadurch ergibt sich eine Einsparung fossiler Energieträger für die Gewinnung und den Transport sowie von Strom bei den Veredlungsprozessen der Rohstoffe, der (Stand 2018) zu etwa 59,8 % aus nicht erneuerbaren Energiequellen wie, Kohle, Gas und Kernenergie besteht [198]. Eine Möglichkeit der Realisierung eines Recyclings ist es, die Einzelteile des zu recycelnden Produktes wieder in neuen Produkten zu verbauen oder diese als Ersatzteile zu verwenden. Man spricht bei diesem Ansatz von einem Produktrecycling (Refabrikation). Bei dieser Recyclingstrategie bleibt die Gestalt des Produktes erhalten. Dadurch ist es möglich, das refabrizierte Produkt, mit geringem Aufwand, für den gleichen Einsatzzweck wiederzuverwenden. So werden gegenüber einem normalen Recycling zusätzlich noch die Energie- und Zeitaufwände eingespart, die dafür nötig wären, den Rohstoff in die Form des jeweiligen Produktes zu bringen. Der Einsatz von Recycling und Refabrikation soll neben dem Ziel des Umweltschutzes idealerweise auch wirtschaftliche Vorteile mit sich bringen. Deshalb ist eine der Hauptaufgaben bei der Realisierung solcher Vorhaben, eine effiziente und wirtschaftliche Umsetzung. Gerade durch die steigende Anzahl von Hybrid- und Elektrofahrzeugen auf den Straßen (Abbildung 75), in denen u. a. Leistungselektronik verbaut ist, drängt sich die Frage auf, wie mit diesen Komponenten eine Refabrikation durchgeführt werden kann. Die Aufgabe der Leistungselektronik in Hybrid- und Elektrofahrzeugen ist die Ansteuerung der elektrischen Maschine, die Diagnose des Antriebs und die Kommunikation mit der Fahrzeugsteuerung [199].

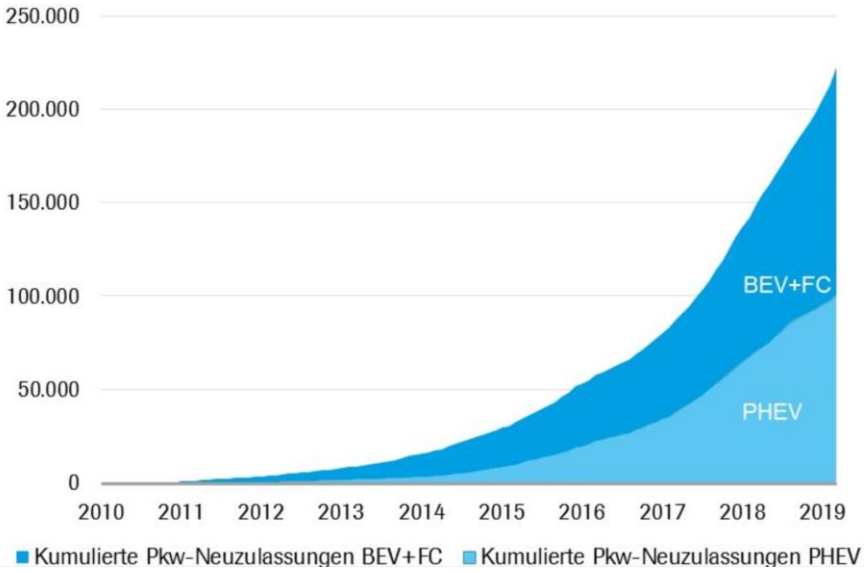


Abbildung 75: Kumulierte Neuzulassungen von Januar 2010 bis Ende März 2019, Quelle: [200], [201]

Die Vielzahl der elektrischen Komponenten wird aktuell allerdings lediglich recycelt und keiner Refabrikation unterzogen. Das Produkt wird im Zuge dessen zerstört, und lediglich die einzelnen Rohmaterialien werden wiederverwendet. Dieses Vorgehen liegt oftmals darin begründetet, dass ein Produktrecycling, mit der damit verbundenen zerstörungsfreien Demontage des Produktes, wirtschaftlich nicht möglich ist. Ein Grund dafür ist, dass die automatisierte Demontage unterschiedlichster Produkte mit den zugehörigen Produktvarianten nicht oder nur unter enormen Aufwand realisierbar ist.

Bei konventionellen Fahrzeugen mit Verbrennungsmotor, werden schon heute eine Vielzahl an Komponenten einer Refabrikation unterzogen. Hierbei werden in den Werkstätten defekte Komponenten ausgebaut und gesammelt, bevor sie an den OEM zurückgesendet werden. Der Kunde der Werkstatt bekommt sein defektes Teil im Austausch durch ein neues, oder bereits refabriziertes Teil, ersetzt. Der OEM führt anschließend selbst, oder durch einen Dienstleister, die Refabrikation der Komponenten durch. Die erhaltenen Komponenten werden abschließend mit einer erneuten Garantie belegt und als Ersatzteile wieder an die Werkstätten veräußert, um einen zweiten Lebenszyklus zu durchlaufen [54]. Beispiele für solche Teile sind Getriebe, Motoren, Starter, Steuergeräte, usw. Der ökologische Vorteil der Refabrikation von Automobilkomponenten wurde in einer Studie der Robert Bosch GmbH am Beispiel eines Starters aus dem Automobilbereich untersucht [202]. Im Rahmen dieser Studie konnten erhebliche ökologische Vorteile nachgewiesen werden, wie in Abbildung 76 zu sehen ist. Die Einsparung an Materialien wird mit 88 % und die Einsparungen bezogen auf CO_2 Emissionen bzw. Energie 53 % respektive 56 % beziffert. Bezogen auf die Logistik ergibt sich eine Einsparung von ca. 2 %.

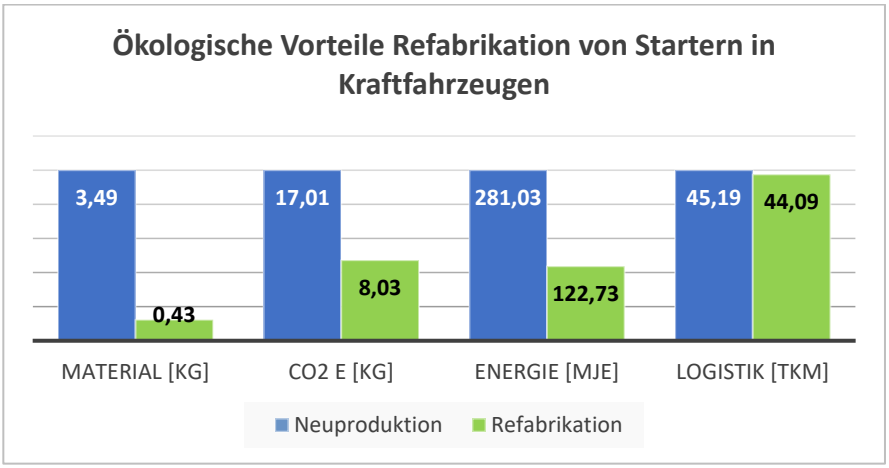


Abbildung 76: Ökologische Vorteile am Beispiel der Refabrikation von Startern der Robert Bosch GmbH [202].

Die jährlichen, positiven Effekte der Refabrikation bzgl. der Einsparung von Rohmaterialien und CO_2 Emissionen wurde in [203] bezogen auf ganz Europa sowie verschiedene Industriesektoren untersucht und in Zahlen gefasst (siehe Tabelle 17). Die Werte beziehen sich auf Europa.

Sektor	Rohstoff ('000t)	CO2 Emissionen ('000t)
Luft- und Raumfahrt	136	356
Automobil	902	3298
EEE	150	177
Möbel	76	131
HDOR	855	3458
Maschinen- und Anlagenbau	35	393
Marinebereich	15	40
Medizinisches Equipment	22	58

Eisenbahn	69	344
Gesamt	**2260**	**8255**

Tabelle 17: Jährliche Einsparung von Rohstoffen und CO_2 Emissionen durch Refabrikation in Europa [203]**. HDOR - Heavy-Duty and Off-Road Vehicles; EEE - Electrical and Electronic Equipment**

Durch die beiden zuvor aufgezeigten Studien, welche die ökologischen Vorteile der Refabrikation sowohl auf Makrolevel, bezogen auf spezifisches Produkt, als auch auf Mikrolevel, bezogen auf ganz Europa, betrachten, zeigen sich große Potenziale in diesem Bereich.

Die Refabrikation ist nicht nur durch ökologische Vorteile oder gesetzliche Rahmenbedingungen motiviert, sondern auch durch ökonomische Vorteile. Sowohl für die Anbieter als auch für die Käufer von Refabrikaten ergeben sich ökonomische Vorteile. Refabrikate im Automobilbereich werden für Preise in Höhe von 45 % bis 65 % des Neupreises des Originalteils [204] angeboten. Dabei kostet die Refabrikation vieler mechatronischer Systeme weniger als 50 % des Neupreises des Originalteils [205]. Dass die Refabrikation eine ökonomisch motivierte Branche ist und nicht nur durch ökologische Motive und gesetzliche Vorgaben getrieben ist, zeigt der Blick auf die amerikanische Refabrikations-Branche, die eine Marktgröße von US 43 $ Milliarden (Stand 2011) besitzt und durch keine Gesetze der Politik getrieben ist und somit rein ökonomisch motiviert ist [203], [206], [207]. Auch der europäische Refabrikationsmarkt besitzt eine Größe von 29,8 € Milliarden und ist folgend, in verschiedene Wirtschaftssektoren untergliedert, dargestellt [203]:

Sektor	Umsatz (€Mrd.)	Firmen	Beschäftigte ('000)	Stückzahl ('000)	Anteil (%)
Luft- und Raumfahrt	12,4	1000	71	5160	11,5
Automobil	7,4	2363	43	27286	1,1
EEE	3,1	2502	28	87925	1,1
Möbel	0,3	147	4	2173	0,4
HDOR	4,1	581	31	7390	2,9
Maschinen- und Anlagenbau	1,0	513	6	1010	0,7
Marinebereich	0,1	7	1	83	0,3
Medizinisches Equipment	1,0	60	7	1005	2,8
Eisenbahn	0,3	30	3	374	1,1
Gesamt	**29,8**	**7203**	**194**	**132406**	**1,9**

Tabelle 18: Kennzahlen des Marktes für Refabrikation in Europa [203]

Der *Anteil* gibt an, wie viel Prozent die Refabrikation, gemessen am gesamten Produktionswert innerhalb des jeweiligen Sektors, ausmacht. Das gesamte europäische Marktvolumen der Produktion für die oben genannten Sektoren beträgt 1,500 € Mrd., sodass der Refabrikationsmarkt mit 29,8 € Mrd. daran einen Anteil von 1,9 % besitzt [203].

Zusammengefasst ergeben sich durch Refabrikation folgende ökonomische Vorteile [207]:

- Einsparung eines Großteils der Kosten für ursprüngliche Ressourcen und Energie
- Meist geringere Kosten für die Refabrikation als für die Neuproduktion eines äquivalenten Produktes
- Kunden profitieren von 45 % bis 65 % geringeren Kosten für Refabrikate gegenüber äquivalenten Neuprodukten
- Durch höhere Gewinnspannen sowie strategische Vorteile ergeben sich Wettbewerbsvorteile

Anhang G - Komplexität kombinatorischer Optimierungsprobleme

Ablaufplanungsprobleme gehören der Klasse der kombinatorischen Optimierungsprobleme an. Bei dieser Art von Optimierungsproblemen liegt eine Menge von diskreten Elementen (Operationen, Orte, …) sowie Nebenbedingungen (Bearbeitungsreihefolgen, Ressourcenverfügbarkeit, …), welche die Beziehung zwischen den einzelnen Elementen beschreiben. Um eine gültige Lösung zu erhalten, darf keine der Nebenbedingungen verletzt sein. Ziel bei der Optimierung ist es, aus dieser Menge an diskreten Elementen eine Teilmenge zu finden, die sowohl die vorhandenen Nebenbedingungen einhält als auch bzgl. einer vorgegebenen Kostenfunktion (minimale DLZ, kürzeste Strecke, …) optimal ist. Beispiele für kombinatorische Optimierungsprobleme sind, das bereits erwähnt JSSP, das „Travelling Salesman Problem“ (TSP), oder auch das „Knapsack Problem“ (Rucksackproblem). Viele kombinatorische Optimierungsprobleme gehören der Klasse der NP-vollständigen Probleme an, was bedeutet, dass diese nicht exakt in Polynominalzeit lösbar sind. Zur Darstellung des Einflusses der Problemgröße *n* und der Komplexität des vorliegenden Problems auf die RZ, haben Korte und Vygen eine Gegenüberstellung dieser Parameter in [208] vorgenommen. Diese sind in Tabelle 19 dargestellt. Die aufgeführten Rechenzeiten basieren auf der Annahme, dass jeder elementare Rechenschritt eine Nanosekunde dauert (entspricht ~ 1000 MHz Taktzeit). Als Beispiel für die Komplexität kombinatorischer Optimierungsprobleme wird das TSP angeführt. Bei diesem Problem muss ein Handelsreisender *n* Städte jeweils genau einmal besuchen und hat das Ziel, die hierzu notwendige Gesamtstrecke zu minimieren. Die einfachste Möglichkeit, das Optimum dieses Problems zu finden, ist es, alle Kombinationsmöglichkeiten durchzuführen. So ergeben sich *n!* Möglichkeiten, wobei *n* die Anzahl der Städte ist [208] (S. 1). Es ist zu sehen, dass die RZ, mit steigender Anzahl an Städten *n*, extrem schnell anwächst. Dies führt dazu, dass der komplette Lösungsraum, also alle möglichen Kombinationen, nicht mehr in praktikabler Zeit durchsucht werden kann, um das Problem exakt zu lösen [208].

n	100n*log(n)	$10n^2$	$n^{3,5}$	$n^{log(n)}$	2^n	n!
10	3 µs	1 µs	3 µs	2 µs	1 µ	4 ms
20	9 µs	4 µs	36 µs	420 µs	1 ms	76 Jahre
50	28 µs	25 µs	884 µs	4 s	13 Tage	10^{48} Jahre
100	66 µs	100 µs	10 ms	5 h	$4 * 10^7$ J.	
10^3	1 ms	10 ms	32 s	$4 * 10^7$ J.		
10^4	13 ms	1 s	28 h			
10^6	2 s	3 h	3169 J.			
10^8	266 s	3 Jahre	$3 * 10^{10}$ J.			
10^{12}	46 Tage	$3 * 10^8$ J.				

Tabelle 19: Zeitkomplexitäten nach Korte und Vygen [208] **(S. 7)**

Die Komplexität eines JSSP ist höher als die eines TSP. Bei einem JSSP mit *n* Aufträgen und *m* jeweils zugehörigen Operationen, entspricht die Komplexität dieses Problems $(n!)^m$ [117]. *n*! gibt die Anzahl der möglichen Reihenfolgen der Operationen auf einer einzelnen Maschine an, und *m* gibt die möglichen Kombinationen über alle *m* Maschinen an. Für ein JSSP mit lediglich $n = 6$ und $m = 4$ ergibt sich eine maximale Anzahl von 268.738.560.000 zulässigen Fertigungsplänen. Dies entspricht einer RZ von 8,5 Jahren (1000 MHz Taktzeit), sollten alle Lösungen vollständig enumeriert werden.

Bei der Verwendung eines Optimierungsverfahrens, welches alle Kombinationen enumeriert und die entsprechend beste Lösung auswählt, ist folglich eine äußerst lange RZ erforderlich. Des Weiteren

führt dies zu einer hyperexponentiellen Steigerung der RZ in Abhängigkeit von der Problemgröße [117]. Aus diesem Grund sind Optimierungsalgorithmen notwendig, die nicht lediglich alle möglichen Kombinationen enumerieren, sondern mit einer effizienteren Strategie nach einem optimalen, oder nahe am Optimum liegenden, Ablaufplan suchen.

Anhang H - Mixed Integer Programming Modell für das Job-Shop Scheduling Problem

Das JSSP kann durch unterschiedliche MIP-Modelle formuliert werden. Eines dieser Modelle ist das von Ku und Beck [180], welches das JSSP wie folgt formuliert:

- J: Endlicher Satz aus n Aufträgen
- M: Endlicher Satz aus m Maschinen
- Jeder Auftrag j besitzt eine durchzuführende Operation auf jeder der vorhanden Maschinen M. Die Reihenfolge kann sich zwischen den Aufträgen unterscheiden.
- σ_h^j: h-te Operation von Auftrag j
- σ_m^j: Letzte Operation von Auftrag j
- Jede Maschine kann zu jedem Zeitpunkt nur maximal einen Auftrag bearbeiten
- Die begonnene Bearbeitung einer Operation auf einer Maschine darf nicht unterbrochen werden.
- p_{ij}: Prozesszeit der Operation j auf Maschine i
- x_{ij}: Startzeit (Integer) von Auftrag j auf Maschine i
- z_{ijk}: Entspricht 1, wenn Auftrag j Vorgänger von Auftrag k auf Maschine i ist

$$\min \ C_{\max} \tag{1}$$

$$x_{ij} \geq 0, \qquad \forall j \in J, i \in M \tag{2}$$

$$x_{\sigma_h^j, j} \geq x_{\sigma_{h-1}^j, j} + p_{\sigma_{h-1}^j, j}, \qquad \forall j \in J, h = 2, ..., m \tag{3}$$

$$x_{ij} \geq x_{ik} + p_{ik} - V \bullet z_{ijk}, \qquad \forall j,k \in J,\ j < k, i \in M \tag{4}$$

$$x_{ik} \geq x_{ij} + p_{ij} - V \bullet (1 - z_{ijk}), \qquad \forall j,k \in J,\ j < k, i \in M \tag{5}$$

$$C_{\max} \geq x_{\sigma_m^j, j} + p_{\sigma_m^j, j}, \qquad \forall j \in J \tag{6}$$

$$z_{ijk} \in \{0,1\}, \qquad \forall j,k \in J, i \in M \tag{7}$$

Als Zielfunktion dient die Minimierung der DLZ (1). Die Randbedingung (2) stellt sicher, dass die Startzeit eines jeden Auftrags größer oder gleich 0 ist. Die Randbedingung (3) stellt die Einhaltung der Reihenfolgebedingungen innerhalb der Aufträge sicher. Die Randbedingungen (4) und (5) definieren, dass jede Maschine zu jedem Zeitpunkt nur genau einen Auftrag bearbeiten kann. Die Variable V wird durch $V = \sum_{j \in J} \sum_{i \in M} p_{ij}$ beschrieben, sodass die Endzeit keiner Operation die Summe der Prozesszeiten aller Operationen überschreiten kann. Die Randbedingung (6) beschreibt die DLZ als die Endzeit der letzten, fertigbearbeiteten Operation aller Aufträge.

Für die Lösung solcher MIP-Modelle sind unterschiedliche Solver wie bspw. IBM ILOG CPLEX, GUROBI oder SCIP verfügbar.

Anhang I - Job-Shop Scheduling Beispiel mit genetischem Algorithmus:

Der GA kann zur Optimierung unterschiedlicher Problemstellungen angepasst werden. Häufig wird er auf kombinatorische Optimierungsprobleme angewandt. Das folgende Beispiel soll die Anwendung des GA, mit seinem in Abbildung 18 aufgeführten Ablaufschema, auf JSSP zeigen. Das JSSP Beispiel in Abbildung 8 dient hierbei als zu optimierendes Problem.

Im ersten Schritt wird eine initiale Population, also unterschiedliche Lösungen für das Problem, erzeugt. Diese werden bei dem GA als Chromosom bezeichnet. Hierfür werden bspw. Konstruktionsverfahren genutzt. Die initiale Population mit den drei Chromosomen C_1, C_2 und C_3 für das vorliegende Beispiel, ist in Abbildung 77 zu sehen. Der Ablaufplan jedes Chromosoms wird zum einen als Gantt-Diagramm und zum anderen in Form einer Tabelle dargestellt. Die Tabelle besteht aus drei Zeilen, wobei die obere die einzelnen Operationen in zeitlicher Reihenfolge darstellt. Die mittlere Zeile stellt die Maschine, auf der die Operation ausgeführt wird, dar. Die letzte Zeile stellt die Vorgängeroperation dar. Eine 0 in dieser Zeile bedeutet, dass es für die zugehörige Operation keinen Vorgänger gibt. Die Spalten in dieser Tabelle werden bei dem GA als Gene bezeichnet.

Der zweite Schritt des GA, Evaluierung, besteht in der Bewertung eines jeden Chromosoms durch eine Fitnessfunktion. Im vorliegenden Fall ist dies die DLZ. Die Fitnessfunktion für jedes Chromosom ist ebenfalls in Abbildung 77 aufgeführt.

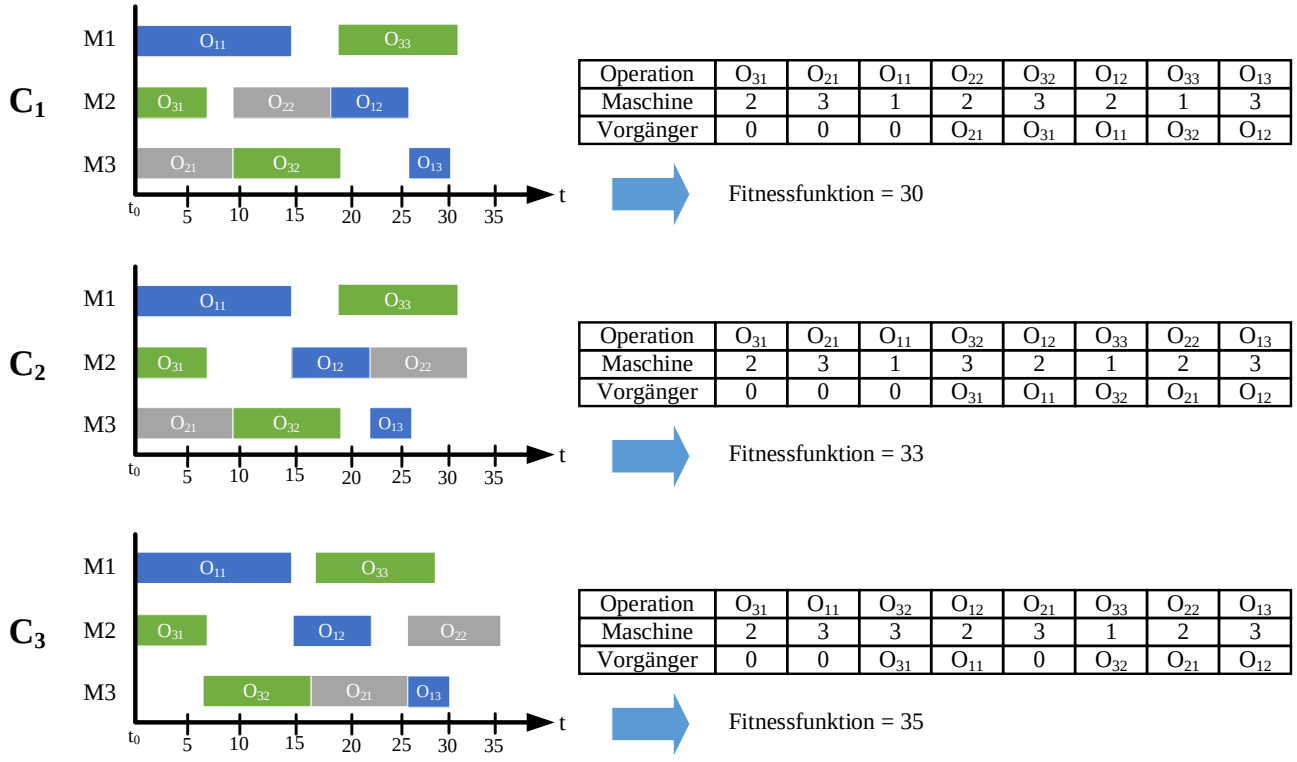


Abbildung 77: Initiale Population mit drei Chromosomen C_1, C_2 und C_3.

Die Selektion ist der dritte Schritt des GA und wählt die Chromosomen aus, welche ihre Gene an die nächste Generation weitervererben sollen. Die ausgewählten Chromosomen werden als Eltern bezeichnet. Unterschiedliche Verfahren können bei der Selektion Anwendung finden. Ein Verfahren ist die Auswahl der Chromosomen mit der höchsten Fitnessfunktion, aber auch eine zufällige Auswahl ist möglich. Das Verfahren der zufälligen Auswahl der Eltern-Chromosomen kann es erlauben, lokalen Optima zu entkommen. Bei dem vorliegenden Beispiel werden C_1 und C_3 als Eltern selektiert.

Im nächsten Schritt Kreuzung, werden die beiden Eltern-Chromosomen miteinander gekreuzt und dadurch neue Chromosomen erzeugt. Diese Chromosomen werden als Kinder-Chromosomen bezeichnet. Die Kreuzung beginnt mit der Auswahl der Gene, welche zwischen den Chromosomen getauscht werden sollen. Für das vorliegende Beispiel werden die Gene 5, 6, 7 und 8 miteinander gekreuzt. Somit ergeben sich für C_1 und C_3 folgende Teilchromosomen:

$$t(C_1) = O_{32}O_{12}O_{33}O_{13}$$
$$t(C_3) = O_{21}O_{33}O_{22}O_{13}$$

Damit keine Operationen in den Chromosomen doppelt vorliegen, werden aus C_1 alle in $t(C_3)$ und aus C_3 alle in $t(C_1)$ vorkommende Operationen gelöscht. C_1 und C_2 sehen nach diesem Schritt wie folgt aus:

$$C_1' = O_{31}O_{11}O_{32}O_{12}$$
$$C_3' = O_{31}O_{11}O_{21}O_{22}$$

Anschließend wird C_1‘ mit $t(C_3)$ und C_3‘ mit $t(C_1)$, sodass die beiden folgenden, neuen Chromosomen ergeben:

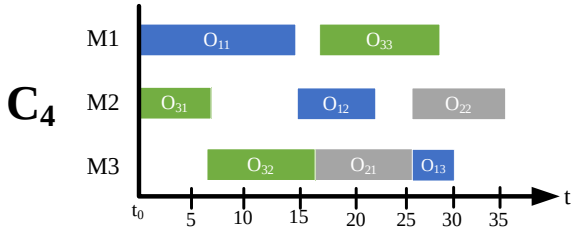
$$C_4(C_1', t(C_3)) = O_{31}O_{11}O_{32}O_{12}O_{21}O_{33}O_{22}O_{13}$$
$$C_5(C_3', t(C_1)) = O_{31}O_{11}O_{21}O_{22}O_{32}O_{12}O_{33}O_{13}$$

Beide Chromosomen enthalten weder doppelte Operationen, noch Verletzungen der Reihenfolgerestriktionen zwischen den einzelnen Operationen innerhalb der Aufträge.

Der letzte Schritt im Ablaufschema des GA ist die Mutation. Diese dient der Vertauschung einzelner Gene innerhalb der Chromosomen. Auch hierfür gibt es unterschiedliche Verfahren. Ein solches Verfahren ist das von Noor et al. [209] modifiziert, und ursprünglich von Lee und Yamakawa [210] vorgestellte *Precedence Preserving Shift (PPS)* Verfahren. Dieses besteht aus den folgenden fünf Schritten:

1. Auswahl einer beliebigen Position *i* innerhalb des Chromosoms
2. Bestimmung der Position *po* der Vorgängeroperation der in Schritt 1 ausgewählten Operation
3. Auswahl einer beliebigen Position *x* zwischen Position *i* und *po*
4. Operation an Position *i* an Position *x* verschieben
5. Vorige an Position x befindliche Operation und alle folgenden Operationen um eine Position nach vorne verschieben

Das modifizierte PPS Verfahren wird im vorliegenden Beispiel auf Chromosom C4 angewandt. Die Position i wird auf das Gen mit O_{33} festgelegt (Abbildung 78). Dadurch ergibt sich für die Position *po* das Gen mit Operation O_{32}. Für Position x wird das Gen der Operation O_{12} ausgewählt.

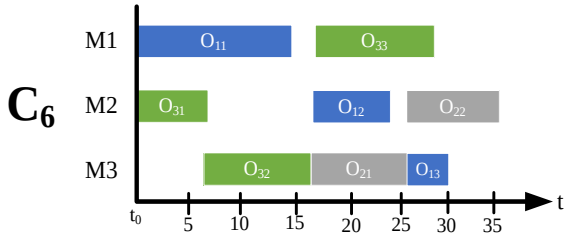


			po	*x*		*i*		
Operation	O_{31}	O_{11}	O_{32}	O_{12}	O_{21}	O_{33}	O_{22}	O_{13}
Maschine	2	3	3	2	3	1	2	3
Vorgänger	0	0	O_{31}	O_{11}	0	O_{32}	O_{21}	O_{12}

Fitnessfunktion = 35

Abbildung 78: GA-Beispiel nach dem Prozessschritt Mutation.

Nach der Durchführung der letzten beiden Schritte des PPS-Verfahrens, ergibt sich das in Abbildung 79 zu sehende Chromosom C_6 mit dem entsprechenden, abgebildeten Ablaufplan. Es ist zu erkennen, dass diese Mutation keine Verbesserung der Fitnessfunktion mit sich gezogen hat.



Operation	O_{31}	O_{11}	O_{32}	O_{33}	O_{12}	O_{21}	O_{22}	O_{13}
Maschine	2	3	3	1	2	3	2	3
Vorgänger	0	0	O_{31}	O_{11}	0	O_{32}	O_{21}	O_{12}

Fitnessfunktion = 35

Abbildung 79: Neues Chromosom C6 nach dem Prozessschritt der Mutation

Nach jeder Iteration der GA Prozessschritte wird die komplette Population bzgl. der Fitnessfunktion untersucht und das Chromosom mit der besten Fitnessfunktion markiert. Die Prozessschritte Evaluation, Selektion, Kreuzung und Mutation werden solange wiederholt, bis eine vorab definierte Abbruchbedingung erfüllt ist. Dadurch wird der Lösungsraum des vorliegenden Problems schrittweise durchsucht.

Als Abbruchbedingung kann bspw. eine bestimmte Anzahl an Iterationen, eine bestimmte Zeit, oder das Erreichen eines definierten Wertes der Fitnessfunktion sein.

Anhang J - IBM ILOG CP Optimizer

CP Optimizer ist ein ursprünglich von *ILOG* entwickelter und später von *IBM* übernommener CP Solver für kombinatorische Optimierungsprobleme. CP Optimizer ist Bestandteil von *IBM ILOG CPLEX Optimization Studio*, einem Tool zur Entwicklung und Lösung von Optimierungsproblemen. *OPL (Optimization Programming Language)* wird darin als Sprache genutzt und umfasst verschiedene Konstanten, Variablen, Ausdrücke und Randbedingungen zur Problembeschreibung. Einige dieser wurden bereits in Kapitel 4.1.1genauer erläutert. Die mit OPL formulierten Probleme können nicht nur mit dem Solver CP Optimizer, sondern auch mit *CPLEX*, einem Solver für Methoden der mathematischen Optimierung, gelöst und optimiert werden. Neben der Nutzung von Optimization Studio sind auch APIs für JAVA, C#, C++, Python und andere Programmiersprachen vorhanden. Dies erlaubt es, die optimierten Lösungen in anderen Softwaremodulen zu nutzen. Der grobe Ablauf zum Lösen eines Problems mit CP Optimizer ist in Abbildung 80 dargestellt. Zuerst wird das Optimierungsproblem in einer der zuvor aufgezählten Programmiersprachen als CP-Modell definiert. Das anschließende Lösen des kombinatorischen Problems erfolgt entweder auf einem lokalen Rechner oder in der IBM Cloud.

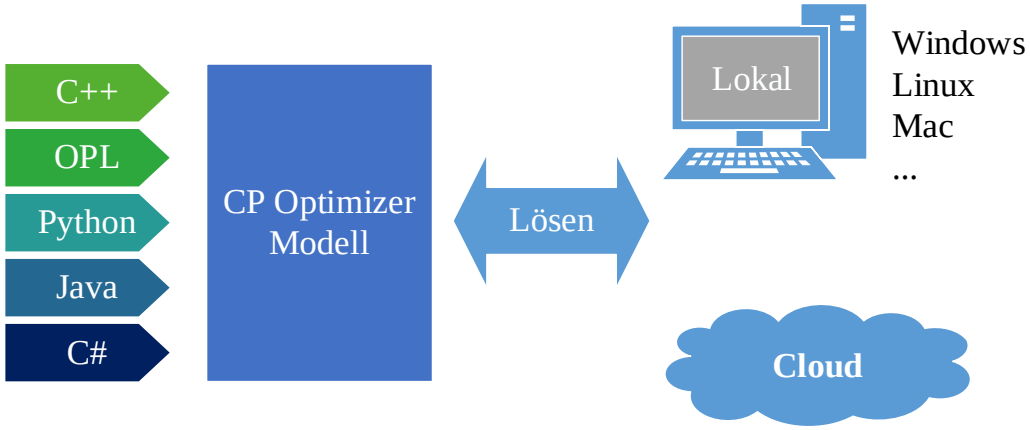


Abbildung 80: CP Optimizer Ansatz nach [179]

CP Optimizer führt die vier, in Abbildung 81 dargestellten, Schritte zur Lösungsfindung durch [179]. Der Solver führt, bevor die eigentliche Lösung und anschließende Optimierung beginnt, einen als *Presolve* bezeichneten Prozess durch. Das CP-Modell wird hierbei unter Anwendung sogenannter Persolve Regeln umformuliert. Ziel ist es das Modell für die folgenden Schritte effizienter zu gestaltet.

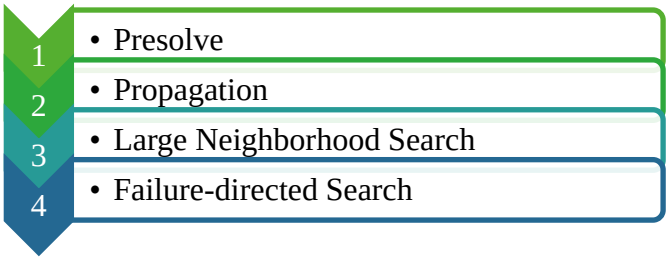


Abbildung 81: Lösungsschritte innerhalb von CP Optimizer.

Der Prozessschritt *Propagation* dient der Repräsentation des vorliegenden Problems durch ein logisches und zeitliches Netzwerk. Das CP-Modell wird zur Erstellung des logischen Netzwerks zuerst als ein 2-SAT (2- satisfiability; Deutsch: 2-Erfüllbarkeiten) Problem definiert. Ziel des 2-SAT Problems ist die Wertezuordnung von Variablen, die zwei mögliche Werte annehmen können, um ein System

von Randbedingungen auf Variablenpaaren zu erfüllen. Hierbei kann jede, problembeschreibende Intervallvariable zwei mögliche Werte annehmen:

- True (wahr; In der Lösung des CP-Modells vorhanden)
- False (falsch; In der Lösung des CP-Modells nicht vorhanden)

Die vorhandenen Randbedingungen zwischen den einzelnen Intervallvariablen werden bei dem 2-SAT Problem durch aussagenlogische Elemente definiert. Grundelemente der Aussagenlogik sind:

$\wedge \rightarrow UND$
$\vee \rightarrow ODER$
$\forall \rightarrow EXKLUSIVES\ ODER$
$\neg \rightarrow NEGATION$

Folgende Aussagenlogik könnte bspw. für die Intervallvariablen x_1, x_2 und x_3 gelten:

$$(x_1 \vee x_2) \wedge (\neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_3) \wedge (x_3 \vee \neg x_2)$$

Zur Lösung des 2-SAT Problems werden alle Wertepaare gesucht, unter welchen diese Aussagenlogik wahr ist.

Das logische Netzwerk aggregiert alle binären Randbedingungen für das Vorhandensein von Intervallvariable des 2-SAT Problems und stellt diese in Form eines Implikationsgraphs dar. Hierbei handelt es sich um einen gerichteten Graphen, in dem für jede Intervallvariable jeweils ein Knoten für deren Vorhandensein sowie für deren Negation vorhanden ist. Hierfür sind zuerst alle Implikationen des vorliegenden, durch Aussagenlogik beschriebenen, Problems zu definierten. Alle in Klammern befindliche Wertepaare des vorherigen Beispiels müssen durch die UND Verbindungen unter diesen wahr sein, damit die gesamte Aussagenlogik wahr ist. Folgende Implikationen ergeben sich dadurch:

$$\begin{array}{cccc}
(x_1 \vee x_2) & \wedge\ (\neg x_2 \vee \neg x_3) & \wedge\ (\neg x_1 \vee x_3) & \wedge\ (x_3 \vee \neg x_2) \\
\downarrow & \downarrow & \downarrow & \downarrow \\
\neg x_1 \rightarrow x_2 & x_2 \rightarrow \neg x_3 & x_1 \rightarrow x_3 & \neg x_3 \rightarrow \neg x_2 \\
\neg x_2 \rightarrow x_1 & x_3 \rightarrow \neg x_2 & \neg x_3 \rightarrow \neg x_1 & x_2 \rightarrow x_3
\end{array}$$

Die Implikation $\neg x_1 \rightarrow x_2$ ist bspw. so zu deuten, dass wenn x_1 false ist, x_2 zwei true sein muss. Diese Implikation ergibt sich durch die geltende ODER-Logik zwischen x_1 und x_2. Eine Überführung dieser Implikationen in einen Implikationsgraphen ist in Abbildung 82 dargestellt. Die Implikationen zwischen den Intervallvariablen sowie deren beiden möglichen Werte, sind dabei durch Pfeile repräsentiert.

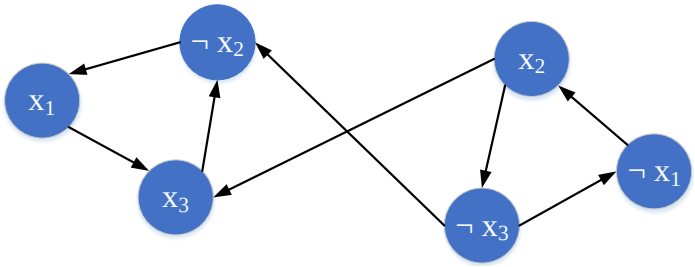


Abbildung 82: Beispielhafter Implikationsgraph zur graphischen Darstellung eines 2-SAT Problems.

Ist eine der Intervallvariablen in der Lage ausgehend von Ihrem True-Knoten ihren False-Knoten zu erreichen und umgekehrt führt dies zu einem Widerspruch. Das Problem ist in diesem Fall nicht lösbar. Das Beispiel in Abbildung 83 soll dies verdeutlichen. Der darin abgebildet Implikationsgraph führt zu einer widersprüchlichen Aussage da $x_1 \wedge \neg x_1$ immer eine falsche Aussage ist.

Abbildung 83: Widerspruch innerhalb eines Implikationsgraphen.

Der Implikationsgraph in Abbildung 82 besitzt für keine der Intervallvariablen x_1, x_2 und x_3 einen solchen Widerspruch, sodass das Problem lösbar (erfüllbar) ist. Dieser kann somit eine, oder mehrere Lösungen besitzen. Alle Pfade innerhalb des Graphs die jeweils genau einen Wert für jede Intervallvariable x_k umfassen und stark miteinander verbunden sind (*strongly connected components*), stellen gültige Lösungen dar. In einem gerichteten Graphen G, der möglicherweise selbst nicht stark verbunden ist, gilt ein Paar von Scheitelpunkten u und v als stark miteinander verbunden, wenn zwischen ihnen ein Pfad in jeder Richtung vorhanden ist. Bezogen auf Abbildung 84 stellen die Knoten x_1, $\neg x_2$ und x_3 nach der vorherigen Definition stark verbundenen Komponenten dar. Werden in darin die Scheitelpunkte $u = x_1$ und $v = x_3$ angenommen, ist sowohl ein Pfad von u nach v als auch von v nach u vorhanden (rote Pfeile).

Die beiden möglichen Lösungen des beispielhaften 2-SAT Problems sind in Abbildung 84 aufgezeigt. Lösung 1 besteht aus allen durch rote Pfeile verbundene Knoten und Lösung 2 aus allen durch grüne Pfeile verbundenen Knoten. Bei beiden Lösungen handelt es sich um stark verbundene Paare. Zudem umfassen sie jeweils genau einen Wert für jede Intervallvariable x_k.

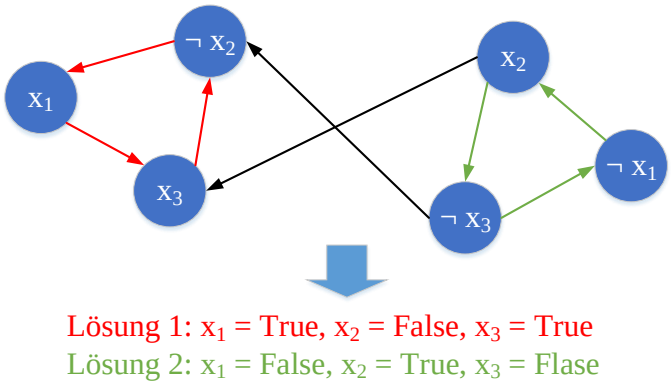


Abbildung 84: Mögliche Lösungen des beispielhaften, als Implikationsgraphen dargestellten, 2-SAT Problems.

Innerhalb des zeitlichen Netzwerks werden alle Reihenfolgerandbedingungen aggregiert und in Form eines Spatio-Temporal Network (STN) repräsentiert. Das STN kann als eine dynamische Struktur von

Abhängigkeiten verstanden werden, die Verbindungen zwischen räumlichen Standorten in verschiedenen Zeitabschnitten enthält. Diese Struktur kann sich im Laufe der Zeit ändern und in Form eines Graphen dargestellt werden [211]. Die Knoten repräsentieren den Start oder das Ende der (optionalen) Intervallvariablen. Die Kanten stellen die minimal einzuhaltende, zeitliche Frist zwischen zwei Knoten dar. Ist eine Intervallvariable in der Lösung vorhanden, wird die zeitliche Domäne des zugehörigen Knotens *t* als ein Bereich $[t_{min}, t_{max}]$ beibehalten.

Für die anschließende Optimierung des so repräsentierten Problems erfolgt die Anwendung von LNS. Die LNS hat das Ziel, schnell zu einer qualitativ guten Lösung zu gelangen. Alternativ kann hierfür auch ein auf GA basierendes Verfahren genutzt werden. Eine vollständige Suche, mit dem Ziel das globale Optimum zu finden, kann durch die Nutzung des Verfahrens *Failure-directed Search* erfolgen. Dieses wird im Anschluss an die LNS durchgeführt, wenn diese beginnt, Probleme bei der weiteren Verbesserung des Problems zu haben. Für eine genauere Erläuterung der Funktionsweise der Failure-directed Search wird auf [212] verwiesen.

Anhang K - Übersicht State-of-the-Art bzgl. SAMF in JSSP- und FJSSP-Umgebung:

Artikel	Methode	Dynamik	FJSSP	JSSP
Bilge und Ulusoy [158]	Non-linear mixed integer programming model	O	O	✓
Nageswararao et al. [159]	Binary Particle Swarm Vehicle Heuristic Algorithm (BPSVHA)	O	O	✓
Erol et al. [183]	MAS	O	O	✓
Mousavi et al. [161]	Hybrider Algorithmus bestehend aus einem Genetischen Algorithmus sowie einer Partikelschwarmoptimierung	O	O	✓
Chaudhry et al. [162]	Genetischer Algorithmus	O	O	✓
Fontes and Homayouni [213]	MILP	O	O	✓
Lacomme et al. [164]	Memetischer Algorithmus	O	O	✓
Fauadi und Murata [165]	Binary Paricle Swarm Optimization	O	O	✓
Somanath et al. [214]	Differential Evolution und Simulated Annealing	O	O	✓
Kanakavalli et al. [215]	Fuzzy Heuristik Algorithmus	O	O	✓
Ham [175]	CP	O	O	✓
Chawla et al. [216]	Grey Wolf Algorithmus	O	O	✓
Mousavi et al. [161]	Hybrider Algorithmus aus GA und PSO	O	O	✓
Chawla et al. [217]	Modified Memetic Particle Swarm Optimization Algorithm (MMPSO)	O	O	✓
Pandey [218]	ARENA Simulation	O	O	✓
Tang [219]	Hormone Regulation Algorithmus	O	O	✓
Deroussi und Norre [166]	Iterative Local Search (ILS)	O	✓	O
Zhang et al. [87]	Hybrider Algorithmus bestehender aus einer Kombination von Genetischem Algorithmus und Tabu-Suche	O	✓	O
Zhang et al. [171]	Hybrider Algorithmus bestehender aus einer Kombination von einem GA, einer Shifting Bottleneck Prozedur und Tabu-Suche	O	✓	O
Kumar et al. [167]	Hybrider Algorithmus bestehend auf einem Differential Evolution Algorithmus, einer Vehicle Assignment Heuristic und einer Machine Selection Heuristic	O	✓	O
Sahin et al. [168]	MAS	✓	✓	O
Lin et al. [170]	Simulationsbasierte Optimierung	✓	✓	O
Deroussi et al. [220]	Hybrider Algorithmus bestehend aus Partikelschwarmoptimierung und stochastischer Lokaler Suche	O	✓	O
Nouri et al. [173]	Hybrider Algorithmus bestehender aus einer Kombination von Genetischem Algorithmus und Tabu-Suche	O	✓	O
Homayouni und Fontes [174]	Mixed Integer Linear Programming model	O	✓	O

Ham [176]	CP	O	✓	O

Tabelle 20: Übersicht bzgl. State-of-the-Art zur SAMF.

Anhang L – BI zur SAMF in einer JSSP-Umgebung nach Bilge und Ulusoy [158]

Auftragssätze zur SAMF ohne alternativen Maschinen im Kontext JSSP nach Bilge und Ulusoy [158]**:**

Job Set 1
Job 1: M1(8); M2(16); M4(12)
Job 2: M1(20); M3(10); M2(18)
Job 3: M3(12); M4(8); M1(15)
Job 4: M4(14); M2(18)
Job 5: M3(10); M1(15)

Job Set 2
Job 1: M1(10); M4(18)
Job 2: M2(10); M4(18)
Job 3: M1(10); M3(20)
Job 4: M2(10); M3(15); M4(12)
Job 5: M1(10); M2(15); M4(12)
Job 6: M1(10); M2(15); M4(12)

Job Set 3
Job 1: M1(16); M3(15)
Job 2: M2(18); M4(15)
Job 3: M1(20); M2(10)
Job 4: M3(15); M4(10)
Job 5: M1(8); M2(10); M3(15); M4(17)
Job 6: M2(10); M3(15); M4(8); M1(15)

Job Set 4
Job 1: M4(11); M1(10); M2(7)
Job 2: M3(12); M2(10); M4(8)
Job 3: M2(7); M3(10);M1(9); M3(8)
Job 4: M2(7); M4(8); M1(12); M2(6)
Job 5: M1(9); M2(7); M4(8); M2(10); M3(8)

Job Set 5
Job 1: M1(6); M2(12); M4(9)
Job 2: M1(18); M3(6); M2(15)
Job 3: M3(9); M4(3);M1(12)
Job 4: M4(6); M2(15)
Job 5: M3(3); M1(9)

Job Set 6
Job 1: M1(9); M2(11); M4(7)
Job 2: M1(19); M2(20); M4(13)
Job 3: M2(14); M3(20); M4(9)
Job 4: M2(14); M3(20); M4(9)
Job 5: M1(11); M3(16); M4(8)
Job 6: M1(10); M3(12); M4(10)

Job Set 7
Job 1: M1(6); M4(6)
Job 2: M2(11); M4(9)
Job 3: M2(9); M4(7)
Job 4: M3(16); M4(7)
Job 5: M1(9); M3(18)
Job 6: M2(13); M3(19); M4(6)
Job 7: M1(10); M2(9); M3(13)
Job 8: M1(11); M2(9); M4(8)

Job Set 8
Job 1: M2(12); M3(21); M4(11)
Job 2: M2(12); M3(21); M4(11)
Job 3: M2(12); M3(21); M4(11)
Job 4: M2(12); M3(21); M4(11)
Job 5: M1(10); M2(14), M3(18); M4(9)
Job 6: M1(10); M2(14), M3(18); M4(9)

Job Set 9
Job 1: M3(9); M1(12); M2(9); M4(6)
Job 2: M3(16); M2(11); M4(9)
Job 3: M1(21); M2(18); M4(7)
Job 4: M2(20); M3(22); M4(11)
Job 5: M3(14); M1(16), M2(13); M4(9)

Job Set 10
Job 1: M1(11); M3(19); M2(16); M4(13)
Job 2: M2(21); M3(16); M4(14)
Job 3: M3(8); M2(10); M1(14); M4(9)
Job 4: M2(13); M3(20); M4(10)
Job 5: M1(9); M3(16), M4(18)
Job 6: M2(19); M1(21), M3(11); M4(15)

Abbildung 85: Daten bzgl. der als Testproblem genutzten Auftragssätze (Job Sets) für *t/p*-Verhältnis > 0,25 [158].

Transportzeitenmatrix der BI nach Bilge und Ulusoy für *t/p*-Verhältnis > 0,25 [158]**:**

	L/U	M1	M2	M3	M4
Layout 1					
L/U	0	6	8	10	12
M1	12	0	6	8	10
M2	10	6	0	6	8
M3	8	8	6	0	6
M4	6	10	8	6	0
Layout 2					
L/U	0	4	6	8	6
M1	6	0	2	4	2
M2	8	12	0	2	4
M3	6	10	12	0	2
M4	4	8	10	12	0
Layout 3					
L/U	0	2	4	10	12
M1	12	0	2	8	10
M2	10	12	0	6	8
M3	4	6	8	0	2
M4	2	4	6	12	0
Layout 4					
L/U	0	4	8	10	14
M1	18	0	4	6	10
M2	20	14	0	8	6
M3	12	8	6	0	6
M4	14	14	12	6	0

Tabelle 21: Transportzeitenmatrix für das Testproblem nach Bilge und Ulusoy für *t/p*-Verhältnis > 0,25 [158].

Transportzeitenmatrix der BI nach Bilge und Ulusoy für *t/p*-Verhätnis < 0,25 [158]:

	L/U	M1	M2	M3	M4
Layout 1					
L/U	0	3	4	5	6
M1	6	0	3	4	5
M2	5	3	0	3	4
M3	4	4	3	0	3
M4	3	5	4	3	0
Layout 2					
L/U	0	2	3	4	3
M1	3	0	1	2	1
M2	4	6	0	1	2
M3	3	5	6	0	1
M4	2	4	5	6	0
Layout 3					
L/U	0	1	2	5	6
M1	6	0	1	4	5
M2	5	6	0	3	4
M3	2	3	4	0	1
M4	1	2	3	6	0
Layout 4					
L/U	0	2	4	5	7
M1	9	0	2	3	5
M2	10	7	0	4	3
M3	6	4	3	0	3
M4	7	7	6	3	0

Tabelle 22: Transportzeitenmatrix für das Testproblem nach Bilge und Ulusoy für *t/p*-Verhältnis < 0,25 [158].

Die jeweilige Transportzeitenmatrix wird in Form einer .data-Datei verwaltet. Diese wird in die Java Applikation zur SAMF eingelesen. Die .data-Datei für Layout 1 der Instanzen mit einem *t/p*-Verhältnis < 0,25 ist in Abbildung 86 zu sehen.

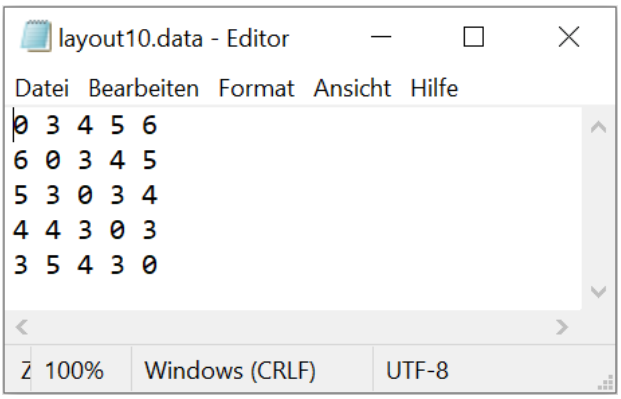


Abbildung 86: Transportzeitenmatrix des Layout 1 der Instanzen mit einem *t/p*-Verhältnis < 0,25 [158] in Form einer .data-Datei.

Anhang M - Aufgabenzuordnung durch das Kontraktnetz-Protokoll in MAS

Das Kontraktnetz-Protokoll dient der Zuordnung von Aufgaben zu den einzelnen Agenten innerhalb eines MAS. Es wurde 1980 von Smith [221] vorgestellt. Die Zuordnung einer Aufgabe erfolgt durch eine feste Vorgehensweise, bei der die betroffenen Agenten entweder die Rolle des Managers oder die des Bieters (*Contractors*) annehmen. Diese Vorgehensweise und die jeweiligen Aufgaben des Managers und des Bieters sind in Abbildung 87 dargestellt. Der Manager startet den Prozess, indem er die Aufgabe an alle, aus seiner Sicht möglichen, Bieter ausschreibt. Diese bewerten nach dem Erhalt der Ausschreibung, ihre Fähigkeiten zur Ausführung der Aufgabe, und senden das Ergebnis in Form eines Angebots an den Manager. Der Manager wählt aus allen erhaltenen Angeboten das Beste aus und erteilt dem zugehörigen Bieter den Zuschlag. Dieser, als *Kontraktor* bezeichnete Bieter, beginnt anschließend mit der Ausführung der Aufgabe. Die Bewertung der Angebote erfolgt oftmals unter der Verwendung von Prioritätsregeln.

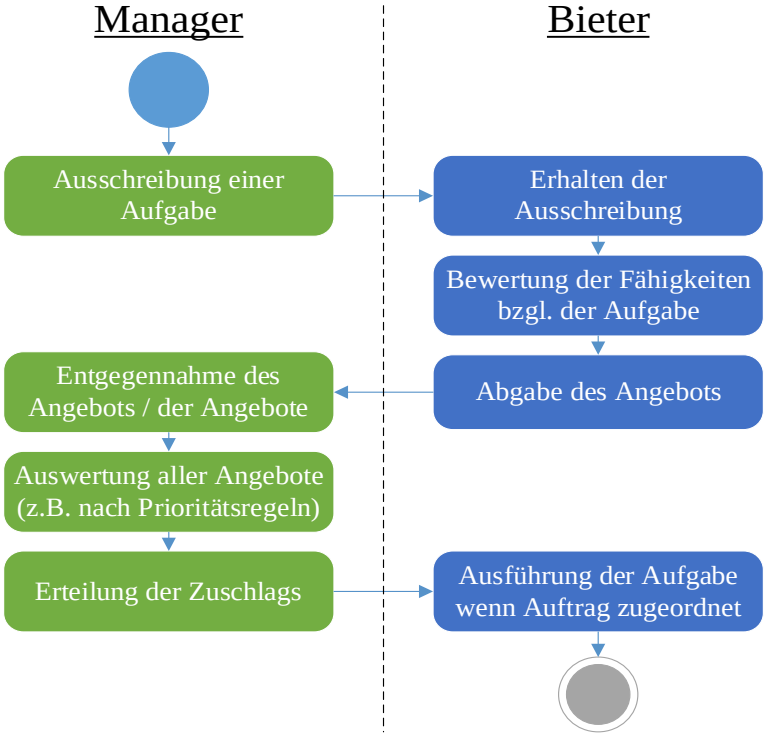


Abbildung 87: Zuordnung einer Aufgabe innerhalb des Kontraktnetz-Protokolls.

Im Falle der Produktionssteuerung kann die Aufgabe des Managers die softwaretechnische Repräsentation eines Auftrags übernehmen. Diese ordnet die Bearbeitung einer Operation einer passenden Maschine zu. Die Rolle der Bieter übernehmen in diesem Fall alle Maschinen, die in der Lage sind, die Bearbeitung dieser Operation durchzuführen. Die Bieter könnten bspw. den frühestmöglichen Bearbeitungsbeginn, oder den frühestmöglichen Fertigstellungszeitpunkt, als Angebot abgeben. Der Manager entscheidet, je nach verwendeter Prioritätsregel, welche Maschine die Bearbeitung der Operation zugeordnet bekommt.

Anhang N – BI zur SAMF in einer FJSSP-Umgebung nach Kumar et al. [167]

Auftragssätze zur SAMF mit alternativen Maschinen im Kontext FJSSP nach Kumar et al. [167]**:**

Job Set 1

Job1: M1(8); M2(16); M4(12)
Alt1: M2(9); M3(14); M1(13)
Alt2: M3(9); M4(17); M2(10)
Job2: M1(20); M3(10); M2(18)
Alt1: M3(18); M1(13); M4(17)
Alt2: M2(21); M4(8); M3(19)
Job3: M3(12); M4(8); M1(15)
Alt1: M4(11); M2(10); M3(14)
Alt2: M1(11); M3(7); M2(17)
Job4: M4(14); M2(18)
Alt1: M1(16); M3(16)
Alt2: M3(16); M1(16)
Job5: M3(10); M1(15)
Alt1: M2(9); M4(16)
Alt2: M4(11); M3(14)

Job Set 2

Job1: M1(10); M4(18)
Alt1: M3(12); M2(16)
Alt2: M2(11); M3(17)
Job2: M2(10); M4(18)
Alt1: M4(8); M1(20)
Alt2: M3(12); M2(16)
Job3: M1(10); M3(20)
Alt1: M3(12); M2(18)
Alt2: M2(8); M4(22)
Job4: M2(10); M3(15); M4(12)
Alt1: M4(11); M1(17); M2(9)
Alt2: M1(13); M4(13); M3(11)
Job5: M1(10); M2(15); M4(12)
Alt1: M4(12); M1(14); M3(11)
Alt2: M2(8); M3(14); M1(15)
Job6: M1(10); M2(15); M3(12)
Alt1: M3(11); M4(13); M2(13)
Alt2: M2(12); M3(14); M4(11)

Job Set 3

Job1: M1(16); M3(15)
Alt1: M3(14); M1(17)
Alt2: M2(15); M4(16)
Job2: M2(18); M4(15)
Alt1: M1(17); M3(16)
Alt2: M3(16); M2(17)
Job3: M1(20); M2(10)
Alt1: M3(19); M4(11)
Alt2: M4(19); M1(11)
Job4: M3(15); M4(10)
Alt1: M4(17); M2(8)
Alt2: M1(17); M3(8)
Job5: M1(18); M2(10); M3(15); M4(17)
Alt1: M3(10); M4(12); M2(13); M1(15)
Alt2: M2(9); M1(8); M4(13); M3(17)
Job6: M2(10); M3(15); M4(8); M1(15)
Alt1: M1(8); M4(14); M2(10); M3(16)
Alt2: M1(9); M2(16); M3(9); M4(14)

Job Set 4

Job1: M4(11); M1(10); M2(7)
Alt1: M3(10); M2(9); M4(9)
Alt2: M2(13); M4(9); M3(6)
Job2: M3(12); M2(10); M4(8)
Alt1: M1(10); M4(11); M3(9)
Alt2: M4(11); M3(9); M2(10)
Job3: M2(7); M3(10); M1(12); M3(8)
Alt1: M4(10); M2(9); M3(8); M1(7)
Job4: M2(7); M4(8); M1(12); M2(6)
Alt1: M3(9); M2(6); M4(10); M1(8)
Alt2: M1(8); M3(9); M2(9); M4(6)
Job5: M1(9); M2(7); M4(8); M2(10); M3(8)
Alt1: M3(8); M4(8); M3(10); M1(9); M2(7)
Alt2: M2(8); M3(8); M1(9); M3(8); M4(9)

Abbildung 88: Erweiterte BI zur SAMF im Kontext FJSSP von Kumar et al. [167] **Teil (1).**

Job Set 5

Job1: M1(6); M2(12); M4(9)
Alt1: M3(7); M4(10); M2(10)
Alt2: M2(7); M3(10); M1(10)
Job2: M1(18); M3(6); M2(15)
Alt1: M3(16); M2(7); M1(16)
Alt2: M2(17); M4(8); M3(14)
Job3: M3(9); M4(3); M1(12)
Alt1: M1(11); M3(4); M2(9)
Alt2: M4(10); M2(5); M3(11)
Job4: M4(6); M2(15)
Alt1: M2(8); M3(13)
Alt2: M3(7); M1(14)
Job5: M3(3); M1(9)
Alt1: M4(5); M2(7)
Alt2: M2(4); M4(8)

Job Set 6

Job1: M1(19); M2(11); M4(7)
Alt1: M3(10); M1(9); M2(8)
Alt2: M4(11); M3(10); M1(6)
Job2: M1(19); M2(20); M4(13)
Alt1: M4(20); M3(18); M2(14)
Alt2: M2(18); M3(22); M1(12)
Job3: M2(14); M3(20); M4(9)
Alt1: M3(12); M4(21); M1(10)
Alt2: M4(15);M1(21);M2(7)
Job4: M2(14); M3(20); M4(9)
Alt1: M4(13); M1(19); M3(11)
Alt2: M3(12); M4(19); M1(12)
Job5: M1(11); M3(16); M4(8)
Alt1: M2(12); M1(14); M3(9)
Alt2: M3(12); M4(13); M2(10)
Job6: M1(10); M3(12); M4(10)
Alt1: M3(9); M4(11); M2(12)
Alt2: M2(9); M1(11); M3(12)

Job Set 7

Job1: M1(6); M4(6)
Alt1: M3(7); M2(5)
Alt2: M4(5); M3(7)
Job2: M2(11); M4(9)
Alt1: M4(10); M3(10)
Alt2: M3(12); M1(8)
Job3: M2(9); M4(7)
Alt1: M3(10); M1(6)
Alt2: M4(8); M3(8)
Job4: M3(16); M4(7)
Alt1: M2(14); M3(9)
Alt2: M1(17); M2(6)
Job5: M1(9); M3(18)
Alt1: M4(11); M2(20)
Alt2: M2(10); M4(17)
Job6: M2(13); M3(19); M4(6)
Alt1: M3(12); M4(18); M1(8)
Alt2: M1(14); M2(17); M3(7)
Job7: M1(10); M2(9); M3(13)
Alt1: M3(11); M1(10); M2(11)
Alt2: M4(11); M3(10); M1(11)
Job8: M1(11); M2(9); M4(8)
Alt1: M2(9); M3(10); M1(9)
Alt2: M4(10); M1(11); M3(7)

Job Set 8

Job1: M2(12); M3(21); M4(11)
Alt1: M1(13); M4(19); M3(12)
Alt2: M3(14); M2(18); M1(12)
Job2: M2(12); M3(21); M4(11)
Alt1: M3(14); M2(20); M1(10)
Alt2: M1(13); M4(19); M3(12)
Job3: M2(12); M3(21); M4(11)
Alt1: M4(13); M1(22); M3(9)
Alt2: M1(10); M4(22); M2(12)
Job4: M2(12); M3(21); M4(11)
Alt1: M3(11); M1(23); M2(10)
Alt2: M4(13); M2(22); M3(9)
Job5: M1(10); M2(14); M3(18); M4(9)
Alt1: M4(13); M3(13); M1(17); M2(8)
Alt2: M3(11); M4(13); M2(16); M1(11)
Job6: M1(10); M2(14); M3(18); M4(9)
Alt1: M2(12); M4(13); M1(16); M3(10)
Alt2: M4(9); M3(15); M2(17); M1(10)

Abbildung 89: Erweiterte BI zur SAMF im Kontext FJSSP von Kumar et al. [167] **Teil (2).**

Job Set 9

Job1: M3(9); M1(12); M2(9); M4(6)
Alt1: M4(10); M3(9); M1(10); M2(7)
Alt2: M2(8); M4(15); M3(8); M1(5)
Job2: M3(16); M2(11); M4(9)
Alt1: M1(14); M3(12); M2(10)
Alt2: M2(18); M4(10); M3(8)
Job3: M1(21); M2(18); M4(7)
Alt1: M3(19); M4(19); M1(8)
Alt2: M4(22); M1(16); M2(8)
Job4: M2(20); M3(22); M4(11)
Alt1: M4(21); M1(20); M3(12)
Alt2: M3(22); M2(21); M1(10)
Job5: M3(14); M1(16); M2(13); M4(9)
Alt1: M2(15); M4(13); M1(14); M3(10)
Alt2: M1(16); M3(14); M4(11); M2(11)

Job Set 10

Job1: M1(11); M3(19); M2(16); M4(13)
Alt1: M2(13); M4(17); M3(14); M1(15)
Alt2: M4(13); M2(18); M1(14); M3(14)
Job2: M2(21); M3(16); M4(14)
Alt1: M3(19); M4(17); M1(15)
Alt2: M1(20); M4(18); M3(13)
Job3: M3(8); M2(10); M1(14); M4(9)
Alt1: M2(10); M4(8); M3(16); M1(7)
Alt2: M1(10); M4(8); M2(15); M3(8)
Job4: M1(13); M3(20); M4(10)
Alt1: M1(15); M4(17); M3(11)
Alt2: M3(11); M1(20); M2(12)
Job5: M1(9); M3(16); M4(18)
Alt1: M3(12); M4(14); M1(17)
Alt2: M4(10); M1(17); M2(16)
Job6: M2(19); M1(21); M3(11); M4(15)
Alt1: M1(16); M3(22); M4(14); M2(14)
Alt2: M4(17); M2(19); M1(13); M3(17)

Abbildung 90: Erweiterte BI zur SAMF im Kontext FJSSP von Kumar et al. [167] **Teil (3).**

Anhang O - CP-Modell zur Beschreibung eines JSSP:

Im Folgenden das CP-Modell eines JSSP vorgestellt. Diese wird zur Durchführung der sequenziellen Ablaufplanung in Kapitel 4.1.1 genutzt.

Variablen:

M: Menge von Maschinen, $\{1, 2, \ldots, m\}$

P: Menge von Aufträgen, $\{1, 2, \ldots, n\}$

S_p: Menge von Operationen des Auftrags p, $\{1, 2, \ldots, k\}$, $p \in P$

b_{sp}: Bearbeitungszeit der Operation s des Auftrags p, $s \in S$, $p \in P$

c_{max}: DLZ

Intervallvariable:

Intervallvariable für jede, zu bearbeitende Operation. Der jeweilige Wert entspricht b_{sp}

interval s_{pi}

Intervallfolgevariable:

Intervallfolgevariable für jede Maschine, welche die Intervallvariablen der jeweiligen auf der Maschine zu bearbeitenden Operationen umfasst

intervalSequenceVar(s_{pi}*)* M_i

Randbedingungen (Constraints):

Einhaltung der Operationsreihenfolge für jeden Auftrag p

endBeforeStart(s_{pi-1}, s_{pi}*)*

Eine Maschine kann zu jeder Zeit nur eine Operation bearbeiten

noOverlap(M_i*)*

Zielfunktion:

Minimierung der DLZ. *s[i]* entspricht einem Array, das alle Operationen eines Auftragssatzes beinhaltet.

minimize max(i in 1..n) endOf(s[i])

Anhang P - CP-Modell zur Beschreibung eines FJSSP:

Im Folgenden das CP-Modell eines FJSSP vorgestellt. Diese wird zur Durchführung der sequenziellen Ablaufplanung in Kapitel 4.1.2 genutzt.

Variablen:

M: Menge von Maschinen, $\{1, 2, \ldots, m\}$

P: Menge von Aufträgen, $\{1, 2, \ldots, n\}$

S_p: Menge von Operationen des Auftrags p, $\{1, 2, \ldots, k\}$, $p \in P$

O_{spm}: Menge von Maschinen aus M für die Bearbeitung der Operation s des Auftrags p, $\{1, 2, \ldots, o\}$, $s \in S$, $m \in M$, $p \in P$

b_{spm}: Bearbeitungszeit der Operation s des Auftrags p auf Maschine m, $s \in S$, $m \in M$, $p \in P$

c_{max}: DLZ

Intervallvariable:

Für jede Bearbeitungsmöglichkeit einer jeden Operation muss eine eigene Intervallvariable definiert werden. Der jeweilige Wert entspricht b_{spm}. Diese Intervallvariable wird als optional definiert, da jede Operation nur von einer der für ihre Bearbeitung in Frage kommenden Maschinen tatsächlich bearbeitet wird.

interval o_{spmi}, *optional*

Intervallfolgevariable:

Intervallfolgevariable für jede Maschine, welche die Intervallvariablen der jeweiligen auf der Maschine zu bearbeitenden Operationen umfasst

intervalSequenceVar(s_{pi}*)* M_i

Randbedingungen (Constraints):

Jede Operation kann alternativ von einer der für ihre Bearbeitung verfügbaren Maschinen bearbeitet werden

alternativ(o_{spm}*)*

Einhaltung der Operationsreihenfolge für jeden Auftrag p

endBeforeStart(s_{pi-1}, s_{pi}*)*

Eine Maschine kann zu jeder Zeit nur eine Operation bearbeiten

noOverlap(M_i*)*

Zielfunktion:

Minimierung der DLZ. *s[i]* entspricht einem Array, das alle Operationen eines Auftragssatzes beinhaltet.

minimize max(i in 1..n) endOf(s[i])

Anhang Q – Detaillierte Simulationsergebnisse

Detaillierte Ergebnisse des Vergleichs zwischen einzellösungs- und mehrpopulationsbasierten Metaheuristiken zur Lösung von CP-Modellen

	JSSP-Umgebung				FJSSP-Umgebung			
	LNS		GA		LNS		GA	
Instanz	DLZ	RZ	DLZ	RZ	DLZ	RZ	DLZ	RZ
EX11	96	0.10	96	0.31	70	1.78	70	11.62
EX21	100	5.71	100	1.06	74	2.02	76	600.05
EX31	99	1.37	99	0.68	69	0.50	69	4.41
EX41	112	336.17	112	3.42	72	1.03	72	100.34
EX51	87	0.10	87	0.25	59	0.64	59	97.78
EX61	118	0.76	118	1.83	79	0.83	88	22.52
EX71	111	70.97	111	105.24	81	1.79	82	600.03
EX81	161	0.04	161	0.04	94	2.17	96	600.02
EX91	116	0.04	116	0.11	82	7.46	84	26.53
EX101	146	0.86	146	0.80	94	0.76	104	600.02
EX12	82	0.03	82	0.11	56	0.46	56	9.15
EX22	76	0.04	76	0.07	62	0.54	63	103.08
EX32	85	0.04	85	0.03	57	1.03	57	9.11
EX42	87	0.15	87	4.50	56	91.78	61	84.68
EX52	69	0.28	69	0.18	49	0.43	49	18.45
EX62	98	0.86	98	0.63	70	0.86	71	600.02
EX72	79	1.32	79	0.93	59	0.86	64	600.02
EX82	151	0.04	151	0.02	82	0.48	83	65.41
EX92	102	0.44	102	0.31	70	2.54	70	14.29
EX102	135	0.05	135	0.46	86	3.45	89	202.55
EX13	84	0.03	84	0.16	62	0.50	62	1.85
EX23	86	0.05	86	0.05	67	0.59	68	71.22
EX33	86	0.06	86	0.15	61	0.38	61	1.55
EX43	89	1.30	89	0.57	62	0.84	66	318.51
EX53	74	0.05	74	0.18	52	0.78	54	10.70
EX63	103	0.05	103	0.08	74	2.98	75	102.71
EX73	83	2.78	83	3.39	66	1.63	69	600.02
EX83	153	0.03	153	0.02	84	1.60	86	368.35
EX93	105	0.13	105	0.21	74	1.65	74	15.35
EX103	137	0.84	137	1.19	90	1.50	95	600.03
EX14	103	1.23	103	0.18	78	0.87	78	56.10
EX24	108	0.48	108	0.70	84	1.80	86	600.02
EX34	111	2.02	111	181.96	77	0.80	77	9.54
EX44	121	3.00	121	5.59	80	2.04	81	600.01
EX54	96	0.64	96	0.57	64	3.38	67	288.30
EX64	120	9.93	120	0.84	87	0.76	90	600.02
EX74	126	15.38	127	415.87	94	3.16	105	600.02
EX84	163	0.11	163	0.13	102	3.42	107	600.02
EX94	120	0.13	120	0.29	87	2.94	89	600.02
EX104	157	1.86	157	5.10	105	1.94	118	600.04
Ø	**108.4**	**11.49**	**108.4**	**18.45**	**74.3**	**3.9**	**76.8**	**275.4**
Median		**0.36**		**0.39**		**1.26**		**102.9**

Tabelle 23: Vergleich zwischen LNS als einzellösungs- und GA als mehrpopulationsbasierte Metaheuristik zur Lösung von CP-Modellen anhand der simultanen Ablaufplanung von Maschinen und FTS.

Detaillierte Ergebnisse des Vergleichs zwischen simultaner, sequenzieller und selbstorganisierter Ablaufplanung ohne alternative Maschinen im Kontext JSSP:

In jeder Zeile sind alle Ergebnisse für eine BI unter Verwendung des jeweiligen Ansatzes aufgeführt. Um welche BI es sich jeweils handelt, ist der ersten Spalte zu entnehmen. Die darin befindlichen Abkürzungen sind wie folgt zu lesen: Für beide *t/p*-Verhältnisse beginnt die Abkürzung mit „EX“ gefolgt von zwei, bei $t/p > 0{,}25$, respektive drei Zahlen bei $t/p < 0{,}25$. Die erste Zahl repräsentiert in beiden Fällen den verwendeten Auftragssatz und die zweite Zahl das verwendete Layout. Die dritte Zahl, bei allen Instanzen mit $t/p < 0{,}25$ repräsentiert, den Faktor, mit welchem die Prozesszeiten multipliziert wurden. Diese Zahl beträgt *0*, wenn die Prozesszeiten mit zwei, und *1*, wenn die Prozesszeiten mit drei multipliziert wurden.

		Simultan	Sequentiell	Selbstorganisation
Instanz		CP	CP+VAH	MAS
Name	*t/p*	Durchlaufzeit		
EX11	0,59	96	147	130
EX21	0,61	100	128	143
EX31	0,59	99	152	142
EX41	0,91	112	195	198
EX51	0,85	87	135	130
EX61	0,78	118	160	153
EX71	0,78	111	175	129
EX81	0,58	161	182	196
EX91	0,61	116	152	178
EX101	0,55	146	185	188
EX12	0,47	82	109	98
EX22	0,49	76	96	86
EX32	0,47	85	102	114
EX42	0,73	87	149	129
EX52	0,68	69	103	98
EX62	0,54	98	110	123
EX72	0,62	79	111	92
EX82	0,46	151	155	172
EX92	0,49	102	137	123
EX102	0,44	135	162	154
EX13	0,52	84	125	109
EX23	0,54	86	104	98
EX33	0,51	86	110	103
EX43	0,8	89	153	155
EX53	0,74	74	107	109
EX63	0,54	103	106	128
EX73	0,68	83	115	93
EX83	0,5	153	157	172
EX93	0,53	105	141	119
EX103	0,49	137	170	158
EX14	0,74	103	183	168
EX24	0,77	108	161	169
EX34	0,74	111	185	167
EX44	1,14	121	237	242
EX54	1,06	96	166	168
EX64	0,78	120	189	189
EX74	0,97	127	197	156
EX84	0,72	163	218	251
EX94	0,76	120	175	181
EX104	0,69	157	219	246
DLZ Ø		**108,4**	**151,6**	**148,9**
Verbesserung		**28,5%**		**1,7%**

		Simultan	Sequentiell	Selbstorganisation
Instanz		CP	CP+VAH	MAS
Name	*t/p*	Durchlaufzeit		
EX110	0,15	126	141	135
EX210	0,15	148	152	157
EX310	0,15	150	160	154
EX410	0,15	119	158	211
EX510	0,21	102	111	118
EX610	0,16	186	191	204
EX710	0,19	137	150	138
EX810	0,14	292	295	330
EX910	0,15	176	191	191
EX1010	0,14	238	245	269
EX120	0,12	123	134	127
EX220	0,12	143	147	151
EX320	0,12	145	152	144
EX420	0,12	114	147	161
EX520	0,17	100	104	110
EX620	0,12	181	186	196
EX720	0,15	136	144	132
EX820	0,11	287	289	319
EX920	0,12	173	183	187
EX1020	0,11	236	237	266
EX130	0,13	122	136	134
EX230	0,13	146	148	151
EX330	0,13	146	154	129
EX430	0,13	114	153	228
EX530	0,18	99	106	111
EX630	0,14	182	187	198
EX730	0,17	137	146	132
EX830	0,13	288	298	273
EX930	0,13	174	188	187
EX1030	0,12	237	241	266
EX140	0,18	124	152	137
EX241	0,13	217	226	230
EX340	0,18	151	166	155
EX341	0,12	221	236	227
EX441	0,19	172	229	344
EX541	0,18	148	165	158
EX640	0,19	184	193	211
EX740	0,24	137	154	158
EX741	0,16	203	220	206
EX840	0,18	293	300	331
EX940	0,19	175	201	195
EX1040	0,17	240	251	276
DLZ Ø		**173,7**	**186,9**	**196,1**
Verbesserung		**7,0%**		**-5,0%**

Tabelle 24: Vergleich der verschiedenen generellen Verfahren zur Ablaufplanung von Maschinen und FTS anhand der BI aus [149] für: Links: t/p > 0,25 und Rechts: t/p < 0,25.

Detaillierte Ergebnisse des Vergleichs zwischen simultaner, sequenzieller und selbstorganisierter Ablaufplanung mit alternativen Maschinen im Kontext FJSSP:

	Simultan	Sequentiell	Selbstgorganisiert
Instanz	CP	CP+VAH	MAS
Name	Durchlaufzeit		
EX11	70	84	111
EX21	74	116	128
EX31	69	98	114
EX41	72	123	163
EX51	59	94	97
EX61	79	123	138
EX71	81	168	124
EX81	94	138	191
EX91	82	111	138
EX101	94	143	170
EX12	56	75	87
EX22	62	81	88
EX32	57	92	99
EX42	56	91	99
EX52	49	70	78
EX62	70	105	94
EX72	59	99	78
EX82	82	111	149
EX92	70	95	107
EX102	86	117	154
EX13	62	78	91
EX23	67	100	102
EX33	61	98	102
EX43	62	101	123
EX53	52	78	88
EX63	74	115	113
EX73	66	109	95
EX83	84	117	146
EX93	74	103	107
EX103	90	127	150
EX14	78	107	128
EX24	84	134	131
EX34	77	124	128
EX44	80	137	168
EX54	64	108	115
EX64	87	149	144
EX74	94	206	124
EX84	102	167	194
EX94	87	130	138
EX104	105	190	180
DLZ Ø	**74.3**	**115.3**	**124.4**
Verbesserung	**35.6%**		**-7.8%**

Tabelle 25: Detaillierte Simulationsergebnisse für alle 40 BI von Kumar et al. [167] **bzgl. dem Vergleich zwischen simultaner, sequenzieller und selbstorganisierter** [168] **Ablaufplanung.**

Detaillierte Ergebnisse des Vergleichs zwischen dem vorgestellten CP-basierten Optimierungsansatz für die simultane Ablaufplanung gegenüber anderen State-of-the-Art Verfahren in einer JSSP-Umgebung:

Zur Verifikation der Leistungsfähigkeit des vorgestellten CP-Ansatzes erfolgt bei Instanzen, bei denen das Verfahren zur Findung der vorgestellten Lösung länger als eine Sekunde benötigt hat, die Darstellung der besten, innerhalb einer Sekunde RZ gefunden Lösung statt (DLZ < 1s).

			Heuristische Verfahren			Exakte Verfahren		Künstliche Intelligenz		
	Erol	Lin	Zheng	Chaudry		Huang		Groß		
	MAS	SBO	TS	GA		MILP		Constraint Programming		
Instanz	DLZ	DLZ	DLZ	DLZ	RZ	DLZ	RZ	DLZ	RZ	RZ < 1s
EX11	130	96	96	96	7	96	30.58	96	0.10	
EX21	143	100	100	100	113	100	730.77	100	5.71	
EX31	142	100	99	99	34	99	176.83	99	1.37	
EX41	198	114	112	112	193	112	50803.3	112	336.17	115
EX51	130	87	87	87	68	87	136.43	87	0.10	
EX61	153	118	118	118	1260	118	7927.26	118	0.76	
EX71	129	111	111	115	104			111	70.97	127
EX81	196	161	161	161	13	161	27.79	161	0.04	
EX91	178	116	116	116	41	116	22.09	116	0.04	
EX101	188	153	146	150	75	146	7138.1	146	0.86	
EX12	98	82	82	82	6	82	4.34	82	0.03	
EX22	86	76	76	76	18	76	5.44	76	0.04	
EX32	114	85	85	85	15	85	8.3	85	0.04	
EX42	129	86	87	88	54	87	3118.96	87	0.15	
EX52	98	69	69	69	10	69	17.82	69	0.28	
EX62	123	98	98	98	120	98	10.18	98	0.86	
EX72	92	79	79	81	240	79	11915	79	1.32	84
EX82	172	151	151	151	4	151	14.77	151	0.04	
EX92	123	102	102	102	20	102	9.69	102	0.44	
EX102	154	135	135	141	300	135	161.63	135	0.05	
EX13	109	84	84	84	18	84	8.14	84	0.03	
EX23	98	86	86	86	7	86	95.98	86	0.05	
EX33	103	86	86	86	54	86	6.68	86	0.06	
EX43	155	89	89	89	25	89	3997.25	89	1.30	
EX53	109	74	74	74	45	74	83.23	74	0.05	
EX63	128	103	103	104	1200	103	23.33	103	0.05	
EX73	93	82	83	90	300	83	33725.1	83	2.78	89
EX83	172	153	153	153	5	153	14.45	153	0.03	
EX93	119	105	105	105	21	105	10.17	105	0.13	
EX103	158	139	137	140	83	137	290.78	137	0.84	
EX14	168	103	103	103	23	103	27.67	103	1.23	
EX24	169	108	108	108	11	108	3698.61	108	0.48	
EX34	167	110	111	111	18	111	832.66	111	2.02	112
EX44	242	126	121	126	68	121	22554.1	121	3.00	
EX54	168	96	96	96	29	96	176.06	96	0.64	98
EX64	189	120	120	122	12	120	1760.19	120	9.93	122
EX74	156	126	126	130	32			127	15.38	133
EX84	251	163	163	163	38	163	4681.18	163	0.11	
EX94	181	122	120	120	42	120	61.69	120	0.13	
EX104	246	159	157	159	161	157	79885	157	1.86	169
Ø	**148.9**	**108.8**	**108.4**	**109.4**	**122.2**	**107.8**	**6162.9**	**108.4**	**11.5**	
Median					**36**		**116.2**		**0.36**	

Tabelle 26: Vergleich der Ergebnisse verschiedener Optimierungsverfahren für die simultane Ablaufplanung von Maschinen und FTS (*t/p > 0,25*).

	Erol	Chaudhy	Groß		
	MAS	GA	Constraint Programming		
Instanz	DLZ	DLZ	DLZ	RZ	RZ < 1s
EX110	135	126	126	0.02	
EX210	157	148	148	0.02	
EX310	154	150	150	0.07	
EX410	211	119	119	0.05	
EX510	118	102	102	0.02	
EX610	204	186	186	0.12	
EX710	138	137	137	0.07	
EX810	330	292	292	0.03	
EX910	191	176	176	0.08	
EX1010	269	238	238	0.6	
EX120	127	123	123	0.02	
EX220	151	143	143	0.02	
EX320	144	145	145	0.09	
EX420	161	114	114	0.07	
EX520	110	100	100	0.01	
EX620	196	181	181	0.03	
EX720	132	136	136	0.02	
EX820	319	287	287	0.03	
EX920	187	173	173	0.04	
EX1020	266	236	236	15.92	237
EX130	134	122	122	0.02	
EX230	151	146	146	0.02	
EX330	129	146	146	0.01	
EX430	228	114	114	0.02	
EX530	111	99	99	0.01	
EX630	198	182	182	0.08	
EX730	132	137	137	0.03	
EX830	273	288	288	0.03	
EX930	187	174	174	0.04	
EX1030	266	237	237	1.03	238
EX140	137	124	124	0.02	
EX241	230	217	217	0.02	
EX340	155	151	151	0.04	
EX341	227	221	221	0.05	
EX441	344	172	172	0.04	
EX541	158	148	148	0.03	
EX640	211	184	184	0.23	
EX740	158	137	137	0.02	
EX741	206	203	203	0.03	
EX840	331	293	293	0.02	
EX940	195	175	175	0.06	
EX1040	276	240	240	32.06	241
Ø	196.1	173.7	173.7	1.3	
Median				0.03	

Tabelle 27: Vergleich der Ergebnisse verschiedener Optimierungsverfahren für die simultane Ablaufplanung von Maschinen und FTS *(t/p < 0,25)*.

Detaillierte Ergebnisse des Vergleichs zwischen dem vorgestellten CP-basierten Optimierungsansatz für die simultane Ablaufplanung gegenüber anderen State-of-the-Art Verfahren in einer FJSSP-Umgebung:

Instanz	Groß		Kumar et al.		Lin et al.	Sahin et al.
	CP		PDE-1	PDE-2	L-GA$_{OCBA}$	FMAS
	DLZ	RZ	DLZ	DLZ	DLZ	DLZ
EX11	**70**	1.78	74	74	72	111
EX21	**74**	2.02	77	77	110	128
EX31	**69**	0.50	81	79	114	114
EX41	**72**	1.03	73	73	127	163
EX51	**59**	0.64	61	61	91	97
EX61	**79**	0.83	83	**79**	127	138
EX71	**81**	1.79	**81**	**81**	121	124
EX81	94	2.17	**93**	94	160	191
EX91	**82**	7.46	**82**	**82**	124	138
EX101	**94**	0.76	97	97	158	170
EX12	**56**	0.46	59	59	76	87
EX22	**62**	0.54	**62**	63	86	88
EX32	**57**	1.03	70	67	95	99
EX42	**56**	91.78	60	60	103	99
EX52	**49**	0.43	50	50	77	78
EX62	**70**	0.86	72	72	111	94
EX72	**59**	0.86	63	64	97	78
EX82	**82**	0.48	83	83	149	149
EX92	**70**	2.54	71	71	111	107
EX102	**86**	3.45	89	89	146	154
EX13	**62**	0.50	64	64	86	91
EX23	**67**	0.59	**67**	**67**	89	102
EX33	**61**	0.38	70	70	94	102
EX43	**62**	0.84	66	64	102	123
EX53	**52**	0.78	**52**	**52**	77	88
EX63	**74**	2.98	79	75	110	113
EX73	**66**	1.63	68	69	95	95
EX83	**84**	1.60	**84**	**84**	149	146
EX93	**74**	1.65	**74**	**74**	112	107
EX103	**90**	1.50	92	92	148	150
EX14	**78**	0.87	80	80	103	128
EX24	**84**	1.80	87	87	111	131
EX34	**77**	0.80	86	86	122	128
EX44	**80**	2.04	83	83	138	168
EX54	**64**	3.38	70	70	95	115
EX64	**87**	0.76	93	93	130	144
EX74	**94**	3.16	100	100	131	124
EX84	**102**	3.42	107	105	162	194
EX94	**87**	2.94	90	90	122	138
EX104	**105**	1.94	**105**	**105**	171	180
Ø	**74.3**	**3.9**	77.5	77.1	115.0	124.4

Tabelle 28: Vergleich der Ergebnisse verschiedener Optimierungsverfahren für die simultane Ablaufplanung von Maschinen und FTS mit alternativen Maschinen.

Anhang R - Leerfahrtenzeitenmatrix der Modellfabrik

		Zu [sec]						
	Z / V	Lager	Demont. 1	Demont. 2	Inspektion	Reinigung	Aufbereitung	Montage
Von [sec]	Lager	-	61	60	61	57	60	63
	Demont. 1	40	38	59	45	50	41	45
	Demont. 2	49	49	38	54	54	49	59
	Inspektion	41	52	55	-	51	41	50
	Reinigung	34	61	59	60	-	56	61
	Aufbereitung	58	63	64	71	68	-	58
	Montage	43	47	54	46	52	39	38

Tabelle 29: Leerfahrtenzeitenmatrix zwischen den einzelnen Stationen innerhalb der Modellfabrik mit Auf- und Abladevorgang zur Modellierung der Leerfahrten.

		Zu [sec]						
	Z / V	Lager	Demont. 1	Demont. 2	Inspektion	Reinigung	Aufbereitung	Montage
Von [sec]	Lager	-	47	46	47	43	46	49
	Demont. 1	26	-	45	31	36	27	31
	Demont. 2	35	35	-	40	40	35	45
	Inspektion	27	38	41	-	37	27	36
	Reinigung	20	47	45	46	-	42	47
	Aufbereitung	44	49	50	57	54	-	44
	Montage	29	33	40	32	38	25	-

Tabelle 30: Transportzeitenmatrix zwischen den einzelnen Stationen innerhalb der Modellfabrik ohne Auf- und Abladevorgang.

Literaturverzeichnis

[1] A. G. Schuh *et al.*, “Cyber Physical Production Control,” in *Industrial Internet of Things: Cybermanufacturing Systems*, S. Jeschke, C. Brecher, H. Song, and D. B. Rawat, Eds. Cham: Springer International Publishing, 2017, pp. 519–539.

[2] H. Meissner, R. Ilsen, and J. C. Aurich, “Analysis of Control Architectures in the Context of Industry 4.0,” in *Procedia CIRP*, 2017.

[3] T. Dimitrov, *Permanente Optimierung dynamischer Probleme der Fertigungssteuerung unter Einbeziehung von Benutzerinteraktionen*. 2015.

[4] Shiyong Wang, Jiafu Wan, Di Li, and Chunhua Zhang, “Implementing Smart Factory of Industrie 4.0: An Outlook,” *Int. J. Distrib. Sens. Networks*, vol. 2016, 2016.

[5] Kat Truner, “Remanufacturing Associations Agree on International Industry Definition - Automotive Parts Remanufacturers Association,” *Automotive Parts Remanufacturers Association (APRA)*, 2016. [Online]. Available: https://apra.org/news/307672/Remanufacturing-Associations-Agree-on-International-Industry-Definition.htm. [Accessed: 06-May-2019].

[6] H. Kagermann, W. Wahlster, and J. Helbig, *Umsetzungsempfehlungen für das Zukunftsprojekt Industrie 4.0-Abschlussbericht des Arbeitskreises Industrie 4.0*, no. April. Promotorengruppe Kommunikation der Forschungsunion Wirtschaft – Wissenschaf, 2013.

[7] F. Riemensperger, “Industrie 4.0 –wie Sensoren, Big Data und 3D-Druck die Produktion und die Arbeit in der Fabrik verändern,” Berlin, 2016.

[8] D. M. Dilts, N. P. Boyd, and H. H. Whorms, “The evolution of control architectures for automated manufacturing systems,” *J. Manuf. Syst.*, vol. 10, no. 1, pp. 79–93, Jan. 1991.

[9] M. Rohloff, “Decentralized production planning and design of a production management system based on an object-oriented architecture,” *Int. J. Prod. Econ.*, vol. 30–31, no. C, pp. 365–383, 1993.

[10] K. Windt and M. Hülsmann, “Changing Paradigms in Logistics --- Understanding the Shift from Conventional Control to Autonomous Cooperation and Control,” in *Understanding Autonomous Cooperation and Control in Logistics: The Impact of Autonomy on Management, Information, Communication and Material Flow*, M. Hülsmann and K. Windt, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 1–16.

[11] C. Ramsauer, *Dezentrale PPS-Systeme Neue Strukturen bei hoher Innovationsdynamik*. Gabler Verlag, 1997.

[12] J. Franke, J. Merhof, and S. Hopfensitz, “Einsatz von dezentralen Multiagentensystemen: Komplexitätsbeherrschung bei der Produktionsplanung und -steuerung,” *ZWF Zeitschrift für wirtschaftlichen Fabrikbetr.*, vol. 105, pp. 1075–1078, 2010.

[13] O. Bendel, “Agilität • Definition | Gabler Wirtschaftslexikon,” *Springer Gabler Verlag*. [Online]. Available: https://wirtschaftslexikon.gabler.de/definition/agilitaet-99882. [Accessed: 06-May-2019].

[14] H. Lödding, *Verfahren der Fertigungssteuerung - Grundlagen, Beschreibung, Konfiguration*, 3rd ed. Hamburg: Spriner Vieweg, 2016.

[15] B. Park and H. R. Choi, “A Genetic Algorithm for Integration of Process Planning and Scheduling in a Job Shop,” 2006, vol. 3, pp. 647–657.

[16] C. W. Leung, T. N. Wong, K. L. Mak, and R. Y. K. Fung, “Integrated process planning and scheduling by an agent-based ant colony optimization,” *Comput. Ind. Eng.*, vol. 59, no. 1, pp. 166–180, 2010.

[17] K. Brankamp, *Ein Terminplanungssystem für Unternehmen der Einzel- und Serienfertigung*. Physica-Verlag HD, 1973.

[18] J. Schlechtendahl, M. Keinert, F. Kretschmer, A. Lechler, and A. Verl, “Making existing production systems Industry 4.0-ready: Holistic approach to the integration of existing production systems in Industry 4.0 environments,” *Prod. Eng.*, vol. 9, no. 1, pp. 143–148, 2014.

[19] B. Scholz-Reiter and M. Freitag, “Autonomous Processes in Assembly Systems,” *CIRP Ann. - Manuf. Technol.*, vol. 56, no. 2, pp. 712–729, 2007.

[20] N. Willeke, B. Kuhrke, F. Kuschicke, R. Scheffermann, and E. Uhlmann, “Simulativer Vergleich zentraler und dezentraler Steuerungen einer Getriebefertigung,” *Simul. Produktion und Logistik 2017*, pp. 249–258, 2017.

[21] S. Grundstein, S. Schukraft, S. Scholz-Reiter, and M. Freitag, “Evaluation system for autonomous control methods in coupled planning and control systems,” *9th CIRP Conf. Intell. Comput. Manuf. Eng.*, vol. 33, no. July, pp. 121–126, 2015.

[22] S. Schukraft, S. Grundstein, B. Scholz-Reiter, and M. Freitag, “Evaluation Approach for the Identification of Promising Methods to Couple Central Planning and Autonomous Control,” *Int. J. Comput. Integr. Manuf.*, vol. 29, pp. 1–24, 2015.

[23] D. Ouelhadj and S. Petrovic, “A survey of dynamic scheduling in manufacturing systems,” *J. Sched.*, vol. 12, no. 4, pp. 417–431, 2008.

[24] D. Siepmann, “Industrie 4.0 – Grundlagen und Gesamtzusammenhang,” in *Einführung und Umsetzung von Industrie 4.0*, A. Roth, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, p. 49.

[25] C. Albert, C. Fuchs, and R. Thome, “Durchblick im Begriffsdschungel der Business-Software,” *Lehrstuhl für BWL und Wirtschaftsinformatik*, pp. 1–10, 2007.

[26] M. Siepermann, “Supply Chain Management (SCM) • Definition | Gabler Wirtschaftslexikon.” [Online]. Available: https://wirtschaftslexikon.gabler.de/definition/supply-chain-management-scm-49361. [Accessed: 21-Aug-2020].

[27] K.-I. Voigt, “Total Quality Management (TQM) • Definition | Gabler Wirtschaftslexikon.” [Online]. Available: https://wirtschaftslexikon.gabler.de/definition/total-quality-management-tqm-47755. [Accessed: 21-Aug-2020].

[28] H. Holland, “Customer Relationship Management (CRM) • Definition | Gabler Wirtschaftslexikon.” [Online]. Available: https://wirtschaftslexikon.gabler.de/definition/customer-relationship-management-crm-30809. [Accessed: 21-Aug-2020].

[29] K. Windt, F. Böse, and T. Philipp, “Autonomy in production logistics: Identification, characterisation and application,” *Robot. Comput. Integr. Manuf.*, vol. 24, no. 4, pp. 572–578, 2008.

[30] G. Schuh and V. Stich, *Produktionsplanung und -steuerung 1*, vol. 4. Springer-Verlag Berlin Heidelberg, 2012.

[31] W. Dangelmeier, U. Pape, and M. Rüther, “Agentensysteme für das Supply Chain Management,” 2004.

[32] A. Tharumarajah, A. J. Wells, and L. Nemes, “Comparison of emerging manufacturing concepts,” 1998, vol. 1, pp. 325–331 vol.1.

[33] M. Wooldridge, “An Introduction to MultiAgent Systems,” 2002, p. 348.

[34] J. Zhang, *Multi-Agent-Based Production Planning and Control*, 1st ed. Shanghai: WILEY, 2017.

[35] P. Vrba and V. Marík, “Capabilities of Dynamic Reconfiguration of Multiagent-Based Industrial Control Systems,” *Syst. Man Cybern. Part A Syst. Humans, IEEE Trans.*, vol. 40, pp. 213–223, 2010.

[36] P. Leitão, “Agent-Based Distributed Manufacturing Control: A State-of-the-Art Survey,” *Eng. Appl. Artif. Intell.*, vol. 22, pp. 979–991, 2009.

[37] P. Lou, S. K. Ong, and A. Nee, “Agent-based distributed scheduling for virtual job shops,” *Int. J. Prod. Res. - INT J PROD RES*, vol. 48, pp. 3889–3910, 2010.

[38] A. Kouider and B. Bouzouia, “Multi-agent job shop scheduling system based on co-operative approach of idle time minimisation,” *Int. J. Prod. Res. - INT J PROD RES*, vol. 50, pp. 409–424, 2012.

[39] S. Wang, J. Wan, D. Zhang, D. Li, and C. Zhang, “Towards smart factory for industry 4.0: a self-organized multi-agent system with big data based feedback and coordination,” *Comput. Networks*, vol. 101, pp. 158–168, 2016.

[40] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, and M. Hoffmann, “Industry 4.0,” *Bus. Inf. Syst. Eng.*, vol. 6, no. 4, pp. 239–242, Aug. 2014.

[41] S. Grundstein, S. Schukraft, M. Görges, and B. Scholz-reiter, “Interlinking central production planning with autonomous production control 2 Central production planning and,” pp. 326–332.

[42] M. E. Leusin, M. Kück, E. M. Frazzon, M. U. Maldonado, and M. Freitag, “Potential of a Multi-Agent System Approach for Production Control in Smart Factories,” *IFAC-PapersOnLine*, vol. 51, no. 11, pp. 1459–1464, 2018.

[43] S. Luo, G. Luo, and X. Zhao, “Common production process modeling for MES based on multi-Agent,” *IEEE Int. Conf. Ind. Eng. Eng. Manag.*, pp. 1582–1586, 2014.

[44] M. Merdan, A. Zoitl, G. Koppensteiner, and M. Melik-Merkumians, “Adaptive Produktionssysteme durch den Einsatz von autonomen Softwareagenten,” *Elektrotechnik und Informationstechnik*, vol. 129, no. 1, pp. 53–58, 2012.

[45] N. Vojdani, B. Erichsen, and T. Lück, “Using production real time data - An agent-based detailed planning by means of simulation,” *Logist. J.*, pp. 1–8, 2017.

[46] R. M. Lima, R. M. Sousa, and P. J. Martins, “Distributed production planning and control agent-based system,” *Int. J. Prod. Res.*, vol. 44, no. 18–19, pp. 3693–3709, 2006.

[47] B. Scholz-Reiter and H. Höhns, “Selbststeuerung logistischer Prozesse mit Agentensystemen,” *Produktionsplan. und -steuerung*, pp. 745–780, 2006.

[48] N. . He, D. Z. . Zhang, and Q. :. Li, “Agentbased hierarchical production planning and scheduling in make-to-order manufacturing system,” *Int. J. Prod. Econ.*, vol. 149, pp. 117–130, 2014.

[49] R. Cupek, A. Ziebinski, L. Huczala, and H. Erdogan, “Agent-based manufacturing execution systems for short-series production scheduling,” *Comput. Ind.*, 2016.

[50] M. Klein, A. Löcklin, N. Jazdi, and M. Weyrich, “A negotiation based approach for agent based production scheduling,” in *Procedia Manufacturing*, 2018.

[51] S. Markus, “Mit dem Cyber-Physischen Produktionssystem zur Industrie-4.0-Fertigung,” Bubenreuth, 2018.

[52] J. Gross Erdmann, “Development of an autonomous control system for target-optimised use of intralogistics means of transport in a production system for individualised products by,” Stellenbosch University, 2020.

[53] O. Cardin, W. Derigent, and D. Trentesaux, “Evolution of holonic control architectures towards Industry 4.0: A short overview,” *IFAC-PapersOnLine*, 2018.

[54] F. J. Weiland, *Make new again by remanufacturing rebuilding refurbishing*. fernand j. weiland, 2016.

[55] Y. Cui, Z. Guan, C. He, and L. Yue, “Research on remanufacturing scheduling problem based on critical chain management,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 215, no. 1, 2017.

[56] M. L. Junior and M. G. Filho, “Production planning and control for remanufacturing : Literature review Production Planning & Control : The Management of Operations Production planning and control for remanufacturing : literature review and analysis,” *Prod. Plan. Control*, no. June, 2012.

[57] C. Zikopoulos, “Remanufacturing lotsizing with stochastic lead-time resulting from stochastic quality of returns,” *Int. J. Prod. Res.*, vol. 55, pp. 1–23, 2016.

[58] R. Zhang, S. K. Ong, and A. Y. C. Nee, “A simulation-based genetic algorithm approach for remanufacturing process planning and scheduling,” *Appl. Soft Comput.*, vol. 37, pp. 521–532, Dec. 2015.

[59] P. Golinska-dawson, M. Kosacka, and A. Nowak, “Automotive Parts Remanufacturing – Experience of Polish Small Automotive Parts Remanufacturing – Experience of Polish Small Companies,” no. January, 2015.

[60] V. D. R. Guide, “Production planning and control for remanufacturing: industry practice and research needs,” *J. Oper. Manag.*, vol. 18, no. 4, pp. 467–483, 2000.

[61] M. Andrew-Munot, A. Yassin, S. T. Syed Shazali, M. Sawawi, S. J. Tanjong, and N. Razali, “Analysis of production planning activities in remanufacturing system,” *J. Mech. Eng. Sci.*, vol. 12, no. 2, pp. 3548–3565, 2018.

[62] M. Lage Junior and M. Godinho Filho, “Master disassembly scheduling in a remanufacturing system with stochastic routings,” *Cent. Eur. J. Oper. Res.*, vol. 25, no. 1, pp. 123–138, 2017.

[63] F. Ehm, “A data-driven modeling approach for integrated disassembly planning and scheduling,” *J. Remanufacturing*, 2018.

[64] J. Jungbluth, W. Gerke, and P. Plapper, “Recent Progress Toward Intelligent Robot Assistants

for Non-Destructive Recent Progress Toward Intelligent Robot Assistants for Non- Destructive Disassembly," in *2. RACIR – Robotix-Academy Conference for Industrial Robotics 2018*, 2018, 1st ed., no. June, pp. 11–20.

[65] P. He, "Optimization and Simulation of Remanufacturing Production Scheduling under Uncertainties," *Int. J. Simul. Model.*, vol. 17, no. 4, pp. 734–743, 2018.

[66] J. M. Kim, Y. D. Zhou, and D. H. Lee, "Priority scheduling to minimize the total tardiness for remanufacturing systems with flow-shop-type reprocessing lines," *Int. J. Adv. Manuf. Technol.*, vol. 91, no. 9–12, pp. 3697–3708, 2017.

[67] H. Wen, S. Hou, Z. Liu, and Y. Liu, "An optimization algorithm for integrated remanufacturing production planning and scheduling system," *Chaos, Solitons & Fractals*, vol. 105, pp. 69–76, 2017.

[68] Y. Qiu, M. Ni, L. Wang, Q. Li, X. Fang, and P. M. Pardalos, "Production routing problems with reverse logistics and remanufacturing," *Transp. Res. Part E Logist. Transp. Rev.*, vol. 111, pp. 87–100, 2018.

[69] C. Liu *et al.*, "An integrated optimization control method for remanufacturing assembly system," *J. Clean. Prod.*, vol. 248, p. 119261, 2020.

[70] A. Brusaferri, E. Leo, L. Nicolosi, D. Ramin, and S. Spinelli, "Integrated automation system with PSO based scheduling for PCB remanufacturing plants," in *2019 IEEE 17th International Conference on Industrial Informatics (INDIN)*, 2019, vol. 1, pp. 990–995.

[71] K. Gao, L. Wang, J. Luo, H. Jiang, A. Sadollah, and Q. Pan, "Discrete harmony search algorithm for scheduling and rescheduling the reprocessing problems in remanufacturing: a case study," *Eng. Optim.*, vol. 50, no. 6, pp. 965–981, 2018.

[72] G. Gong, Q. Deng, R. Chiong, X. Gong, H. Huang, and W. Han, "Remanufacturing-oriented process planning and scheduling: mathematical modelling and evolutionary optimisation," *Int. J. Prod. Res.*, vol. 0, no. 0, pp. 1–19, 2019.

[73] H. P. Zhang, "Optimization of remanufacturing production scheduling considering uncertain factors," *Int. J. Simul. Model.*, vol. 18, no. 2, pp. 344–354, 2019.

[74] M. Li, T. Li, S. Peng, and Y. Guo, "Batch Scheduling of Remanufacturing Flexible Job Shop for Minimal Electricity- and Time-Cost," in *Recent Advances in Intelligent Manufacturing*, 2018, pp. 300–307.

[75] K. Gao, P. Duan, R. Su, and J. Li, "Bi-objective Water Cycle Algorithm for Solving Remanufacturing Rescheduling Problem," in *Simulated Evolution and Learning*, 2017, pp. 671–683.

[76] J.-M. Yu and D.-H. Lee, "Scheduling algorithms for job-shop-type remanufacturing systems with component matching requirement," *Comput. Ind. Eng.*, vol. 120, pp. 266–278, 2018.

[77] L. Li, C. Li, L. Li, Y. Tang, and Q. Yang, "An integrated approach for remanufacturing job shop scheduling with routing alternatives," *Math. Biosci. Eng.*, vol. 16, no. 4, pp. 2063–2085, 2019.

[78] "Feinplanung in der Produktion: Herausforderungen und Lösungen," *abas ERP*, 2018. [Online]. Available: https://abas-erp.com/de/news/feinplanung-in-der-produktion-herausforderungen-und-loesungen. [Accessed: 21-Jun-2019].

[79] M. Weskamp and T. Wochinger, "Manufacturing Executions Systems: Problemanalyse MES-

Implementierung – Unternehmensebene/Engineering – Computer&AUTOMATION," 2012. [Online]. Available: https://www.computer-automation.de/unternehmensebene/engineering/problemanalyse-mes-implementierung.86524.3.html. [Accessed: 18-Dec-2019].

[80] K. Z. Gao, P. N. Suganthan, T. J. Chua, C. S. Chong, T. X. Cai, and Q. K. Pan, "A two-stage artificial bee colony algorithm scheduling flexible job-shop scheduling problem with new job insertion," *Expert Syst. Appl.*, vol. 42, no. 21, pp. 7652–7663, 2015.

[81] J. Gao, L. Sun, and M. Gen, "A hybrid genetic and variable neighborhood descent algorithm for flexible job shop scheduling problems," *Comput. Oper. Res.*, vol. 35, no. 9, pp. 2892–2907, 2008.

[82] S. V. Kamble, S. U. Mane, and A. J. Umbarkar, "Hybrid Multi-Objective Particle Swarm Optimization for Flexible Job Shop Scheduling Problem," *Int. J. Intell. Syst. Appl.*, vol. 7, no. 4, pp. 54–61, 2015.

[83] J. Q. Li, Q. K. Pan, and Y. C. Liang, "An effective hybrid tabu search algorithm for multi-objective flexible job-shop scheduling problems," *Comput. Ind. Eng.*, vol. 59, no. 4, pp. 647–662, 2010.

[84] J. Q. Li, Q. K. Pan, and K. Z. Gao, "Pareto-based discrete artificial bee colony algorithm for multi-objective flexible job shop scheduling problems," *Int. J. Adv. Manuf. Technol.*, vol. 55, no. 9–12, pp. 1159–1169, 2011.

[85] L.-N. Xing, Y.-W. Chen, P. Wang, Q.-S. Zhao, and J. Xiong, "A Knowledge-Based Ant Colony Optimization for Flexible Job Shop Scheduling Problems," *Appl. Soft Comput.*, vol. 10, no. 3, pp. 888–896, 2010.

[86] M. Yazdani, M. Amiri, and M. Zandieh, "Flexible job-shop scheduling with parallel variable neighborhood search algorithm," *Expert Syst. Appl.*, vol. 37, no. 1, pp. 678–687, 2010.

[87] Q. Zhang, H. Manier, and M.-A. Manier, "A genetic algorithm with tabu search procedure for flexible job shop scheduling with transportation constraints and bounded processing times," *Comput. Oper. Res.*, vol. 39, no. 7, pp. 1713–1723, 2012.

[88] H. E. Nouri, O. Belkahla Driss, and K. Ghédira, "Controlling a Single Transport Robot in a Flexible Job Shop Environment by Hybrid Metaheuristics," *LNCS Trans. Comput. Collect. Intell.*, vol. 28, pp. 93–115, 2018.

[89] B. Ramasubramanian, "Re-entrant Manufacturing Systems," pp. 1–27.

[90] H. Losbichler, C. Eisl, and C. Engelbrechtsmüller, *Handbuch der betriebswirtschaftlichen Kennzahlen: Key Performance Indicators für die erfolgreiche Steuerung von Unternehmen*. Linde, 2015.

[91] M. R. Garey, D. Johnson, and R. Sethi, "The Complexity of Flowshop and Jobshop Scheduling," *Math. Oper. Res. - MOR*, vol. 1, pp. 117–129, 1976.

[92] P. Brandimarte, "Routing and scheduling in a flexible job shop by tabu search," *Ann. Oper. Res.*, vol. 41, no. 3, pp. 157–183, 1993.

[93] M. L. Pinedo, *Scheduling Theory, Algorithms, and Systems*, 5th ed. Springer International Publishing, 2016.

[94] P. Fattahi, M. Saidi Mehrabad, and F. Jolai, "Mathematical modeling and heuristic approaches

to flexible job shop scheduling problems," *J. Intell. Manuf.*, vol. 18, no. 3, pp. 331–342, 2007.

[95] G. K. HUTCHINSON and K. A. PFLUGHOEFT, "Flexible process plans: their value in flexible automation systems," *Int. J. Prod. Res.*, vol. 32, no. 3, pp. 707–719, 1994.

[96] B. L. Maccarthy and J. Liu, "Addressing the gap in scheduling research: A review of optimization and heuristic methods in production scheduling," *Int. J. Prod. Res.*, vol. 31, no. 1, pp. 59–79, 1993.

[97] H. Y. Fuchigami and S. Rangel, "A survey of case studies in production scheduling: Analysis and perspectives," *J. Comput. Sci.*, vol. 25, no. February, pp. 425–436, 2018.

[98] S. R. Lawrence and E. C. Sewell, "Heuristic, optimal, static, and dynamic schedules when processing times are uncertain," *J. Oper. Manag.*, vol. 15, no. 1, pp. 71–82, Feb. 1997.

[99] G. E. Vieira, J. W. Herrmann, and E. Lin, "Rescheduling Manufacturing Systems: A Framework of Strategies, Policies, and Methods," *J. Sched.*, vol. 6, no. 1, pp. 39–62, 2003.

[100] I. N. Pujawan, "Schedule nervousness in a manufacturing system: A case study," *Prod. Plan. Control*, vol. 15, no. 5, pp. 515–524, 2004.

[101] S. V. Mehta and R. M. Uzsoy, "Predictable Scheduling of a Job Shop Environment Subject to Breakdowns," *Robot. Autom. IEEE Trans.*, vol. 14, pp. 365–378, 1998.

[102] J.-C. Billaut, A. Moukrim, and E. Sanlaville, "Introduction to Flexibility and Robustness in Scheduling," *Flex. Robustness Sched.*, pp. 15–33, 2010.

[103] I. Sabuncuoglu and M. Bayız, "Analysis of reactive scheduling problems in a job shop environment," *Eur. J. Oper. Res.*, vol. 126, pp. 567–586, 2000.

[104] K. M. M. A. Bukkur, Shukri M.I., and O. M. Elmardi, "A Review for Dynamic Scheduling in Manufacturing," *Glob. J. Res. Eng.*, vol. 18, no. 5-J, pp. 25–37, 2018.

[105] P. Moratori, S. Petrovic, and J. A. Vázquez Rodríguez, "Match-up approaches to a dynamic rescheduling problem," *Int. J. Prod. Res. - INT J PROD RES*, vol. 50, pp. 261–276, 2012.

[106] L. Mönch, "Autonome und kooperative steuerung komplexer produktionsprozesse mit multi-agenten-systemen," *Wirtschaftsinformatik*, vol. 48, no. 2, pp. 107–119, 2006.

[107] VDMA, "VDMA 66412-40:2019-10 MES im Umfeld von Industrie 4 . 0," 2019.

[108] "ISO 22400-1 - 2014-10 - Beuth.de." [Online]. Available: https://www.beuth.de/de/norm/iso-22400-1/224733845. [Accessed: 03-Feb-2020].

[109] ISO, "ISO 22400-2 - 2014-01 - Beuth.de."

[110] E. H. Hartmann and D. Beese, *TPM: Effiziente Instandhaltung und Maschinenmanagement*. Vahlen, 2013.

[111] IBM, *IBM ILOG CPLEX Optimization studio CP Optimizer user's manual*, 12.9. 2019.

[112] M. Hüftle, "Methoden der Optimierung bei mehrfacher Zielsetzung," 2006.

[113] N. Gunantara, "A review of multi-objective optimization: Methods and its applications," *Cogent Eng.*, vol. 5, no. 1, pp. 1–16, 2018.

[114] J. Zhang, G. Ding, Y. Zou, S. Qin, and J. Fu, "Review of job shop scheduling research and its new perspectives under Industry 4.0," *J. Intell. Manuf.*, vol. 30, no. 4, pp. 1809–1830, 2019.

[115] E. S. Page and L. B. Wilson, *An Introduction to Computational Combinatorics*, vol. 1. 1979.

[116] Y. Pochet and L. A. Wolsey, *Production Planning by Mixed Integer Programming (Springer Series in Operations Research and Financial Engineering)*. Springer Berlin Heidelberg, 2006.

[117] R. Kan and A. Hendrik George, *Machine scheduling problems : classification, complexity and computations /*. Springer US, 1976.

[118] M. C. Gomes, A. P. Barbosa-Póvoa, and A. Q. Novais, "Optimal scheduling for flexible job shop operation," *Int. J. Prod. Res.*, vol. 43, no. 11, pp. 2323–2353, 2005.

[119] Y. Demir and S. Isleyen, "Evaluation of mathematical models for flexible job-shop scheduling problems," *Appl. Math. Model.*, vol. 37, pp. 977–988, 2013.

[120] J. Carlier and E. Pinson, "An Algorithm for Solving the Job-Shop Problem," *Manage. Sci.*, vol. 35, no. 2, pp. 164–176, 1989.

[121] E. L. Demeulemeester and W. S. Herroelen, "A Branch-and-Bound Procedure for the Generalized Resource-Constrained Project Scheduling Problem," *Oper. Res.*, vol. 38, no. 12, pp. 1803–1818, 1992.

[122] W. van Wezel, J. Cegarra, and J.-M. Hoc, "Allocating Functions to Human and Algorithm in Scheduling," in *Behavioral Operations in Planning and Scheduling*, J. C. Fransoo, T. Waefler, and J. R. Wilson, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 339–370.

[123] B. Giffler and G. L. Thompson, "Algorithms for Solving Scheduling Problems," *Oper. Res.*, vol. 8, no. 4, pp. 487–503, 1960.

[124] J. H. BLACKSTONE, D. O. N. T. PHILLIPS, and G. Hogg, "A State-of-the-Art Survey of Dispatching Rules for Manufacturing Job Shop Operations," *Int. J. Prod. Res. - INT J PROD RES*, vol. 20, pp. 27–45, 1982.

[125] J. J. Lavos, J. A. Araúzo, and R. del O. Martinez, "Applications of the Lagrangian Relaxation Method to Operations Scheduling," in *Managing Complexity: Challenges for Industrial Engineering and Operations Management*, Springer International Publishing, 2014, pp. 147–153.

[126] R. Kolisch and R. Padman, "An integrated survey of deterministic project scheduling," *Omega*, vol. 29, pp. 249–272, 2001.

[127] L. Di Gaspero, A. Schaerf, and M. Cadoli, "Local Search Techniques for Scheduling Problems: Algorithms and Software Tools," *... di Mat. e ...*, 2003.

[128] B. Funke, T. Grünert, and S. Irnich, "Local Search for Vehicle Routing and Scheduling Problems: Review and Conceptual Integration," *J. Heuristics*, vol. 11, no. 4, pp. 267–306, Jul. 2005.

[129] E. H. L. Aarts, P. J. M. van Laarhoven, J. K. Lenstra, and N. L. J. Ulder, "A Computational Study of Local Search Algorithms for Job Shop Scheduling," *ORSA J. Comput.*, vol. 6, no. 2, pp. 118–125, 1994.

[130] D. Pisinger and S. Ropke, "Large Neighborhood Search," in *handbook of Metaheuristics*, vol. 272, no. June 2014, 2010.

[131] J. Krause, J. Cordeiro, R. S. Parpinelli, and H. S. A. Lopes, "A Survey of Swarm Algorithms Applied to Discrete Optimization Problems," *Swarm Intell. Bio-Inspired Comput.*, no.

November 2017, pp. 169–191, 2013.

[132] F. Glover, “Future paths for integer programming and links to artificial intelligence,” *Comput. Oper. Res.*, vol. 13, no. 5, pp. 533–549, 1986.

[133] T. El-Ghazali, *Metaheuristics. From Design to Implementation*. 2009.

[134] J. E. Biegel and J. J. Davern, “Genetic algorithms and job shop scheduling,” *Comput. Ind. Eng.*, vol. 19, no. 1–4, pp. 81–91, Jan. 1990.

[135] F. Werner, “A survey of genetic algorithms for shop scheduling problems,” *Math. Res. Summ.*, vol. 2, p. 15, 2017.

[136] A. Colorni, M. Dorigo, V. Maniezzo, and M. Trubian, “Ant System for Job-shop Scheduling,” *Belgian J. Oper. Res. Stat. Comput. Sci.*, vol. 34, no. 1, pp. 39–53, 1994.

[137] H. Nazif, “Solving Job Shop Scheduling Problem Using an Ant Colony Algorithm,” *J. Asian Sci. Res.*, vol. 5, no. 5, pp. 261–268, 2015.

[138] M. Liu, G. Zhou, S. Wang, L. Wang, and Y. Xu, “An effective artificial bee colony algorithm for the flexible job-shop scheduling problem,” *Int. J. Adv. Manuf. Technol.*, vol. 60, no. 1–4, pp. 303–315, 2011.

[139] F. Zhang and J. Li, “An Improved Particle Swarm Optimization Algorithm for Integrated Scheduling Model in AGV-Served Manufacturing Systems,” *J. Adv. Manuf. Syst.*, vol. 17, 2018.

[140] K. C. Udaiyakumar and M. Chandrasekaran, “Application of firefly algorithm in job shop scheduling problem for minimization of Makespan,” *Procedia Eng.*, vol. 97, pp. 1798–1807, 2014.

[141] C. L. Liu, C. C. Chang, and C. J. Tseng, “Actor-critic deep reinforcement learning for solving job shop scheduling problems,” *IEEE Access*, vol. 8, pp. 71752–71762, 2020.

[142] K.-L. Huang and C. Liao, “Ant colony optimization combined with taboo search for the job shop scheduling problem,” *Comput. Oper. Res.*, vol. 35, pp. 1030–1046, 2008.

[143] T. König and P. Hofstedt, “Constraint-basierte Programmiersprachen,” *KI - Künstliche Intelligenz*, vol. 26, no. 1, pp. 47–54, Feb. 2012.

[144] “Constraint Optimization | OR-Tools | Google Developers,” *Google AI*. [Online]. Available: https://developers.google.com/optimization/cp/. [Accessed: 30-Jun-2019].

[145] M. Lage and M. Godinho Filho, “Production planning and control for remanufacturing: Exploring characteristics and difficulties with case studies,” *Prod. Plan. Control*, vol. 27, no. 3, pp. 212–225, 2016.

[146] J. Friedrich, “A Associação de Tecnologias de Transporte e Intralogística é a segunda maior associação setorial no âmbito da Associação Alemã de Fabricantes de Máquinas e Instalações Industriais (VDMA). - VDMA,” 2017. [Online]. Available: https://brazil.vdma.org/article/-/articleview/22991311. [Accessed: 03-May-2019].

[147] M. De Ryck, M. Versteyhe, and F. Debrouwere, “Automated guided vehicle systems, state-of-the-art control algorithms and techniques,” *J. Manuf. Syst.*, vol. 54, no. December 2018, pp. 152–173, 2020.

[148] Mobile Industrial Robots, “MiRFleet | Mobile Industrial Robots.” [Online]. Available: https://www.mobile-industrial-robots.com/de/products/mir-add-ons/mirfleet/. [Accessed: 25-

Mar-2019].

[149] K. AG, “KUKA.NavigationSolution | KUKA AG.” [Online]. Available: https://www.kuka.com/en-gb/products/mobility/navigation-solution. [Accessed: 25-Mar-2019].

[150] BA Systèmes, “AGV Manager: AGV fleet management software | BA Systèmes.” [Online]. Available: https://www.basystemes.com/en/agvs/supervision/. [Accessed: 25-Mar-2019].

[151] DEMATIC, “E’tricc - Cutting Edge AGV Management Software.” [Online]. Available: http://egeminusa.com/automated-guided-vehicles/software/etricc/. [Accessed: 25-Mar-2019].

[152] Gebocermex, “AGV supervision system.” [Online]. Available: https://www.gebocermex.com/en/intralogistic-systems/automated-guided-vehicles/agv-supervision-system-pd-111. [Accessed: 25-Mar-2019].

[153] JBT, “SGV Manager - JBT.” [Online]. Available: https://www.jbtc.com/automated-systems/products-and-applications/products/software-and-services/software/sgv-manager. [Accessed: 25-Mar-2019].

[154] savant automation, “Controls & Software | Savant Automation.” [Online]. Available: http://www.agvsystems.com/controls-software/. [Accessed: 25-Mar-2019].

[155] S. C. Srivastava, A. K. Choudhary, S. Kumar, and M. K. Tiwari, “Development of an intelligent agent-based AGV controller for a flexible manufacturing system,” *Int. J. Adv. Manuf. Technol.*, vol. 36, no. 7–8, pp. 780–797, 2008.

[156] E. Cardarelli, V. Digani, L. Sabattini, C. Secchi, and C. Fantuzzi, “Cooperative cloud robotics architecture for the coordination of multi-AGV systems in industrial warehouses,” *Mechatronics*, vol. 45, pp. 1–13, 2017.

[157] F. Yao, A. Keller, M. Ahmad, B. Ahmad, R. Harrison, and A. W. Colombo, “Optimizing the Scheduling of Autonomous Guided Vehicle in a Manufacturing Process,” *Proc. - IEEE 16th Int. Conf. Ind. Informatics, INDIN 2018*, pp. 264–269, 2018.

[158] Ü. Bilge and G. Ulusoy, “A Time Window Approach to Simultaneous Scheduling of Machines and Material Handling System in an FMS,” *Oper. Res.*, vol. 43, no. 6, pp. 1058–1070, 1995.

[159] M. Nageswararao, K. Narayanarao, and G. Ranagajanardhana, “Simultaneous Scheduling of Machines and AGVs in Flexible Manufacturing System with Minimization of Tardiness Criterion,” *Procedia Mater. Sci.*, vol. 5, pp. 1492–1501, 2014.

[160] R. Erol, C. Sahin, A. Baykasoglu, and V. Kaplanoglu, “A multi-agent based approach to dynamic scheduling of machines and automated guided vehicles in manufacturing systems,” *Appl. Soft Comput. J.*, vol. 12, no. 6, pp. 1720–1732, 2012.

[161] M. Mousavi, H. J. Yap, S. N. Musa, F. Tahriri, and S. Z. Md Dawal, “Multi-objective AGV scheduling in an FMS using a hybrid of genetic algorithm and particle swarm optimization,” *PLoS One*, vol. 12, no. 3, pp. 1–24, 2017.

[162] I. A. Chaudhry, S. Mahmood, and M. Shami, “Simultaneous scheduling of machines and automated guided vehicles in flexible manufacturing systems using genetic algorithms,” *J. Cent. South Univ. Technol.*, no. October 2011, 2011.

[163] D. B. M. M. Fontes and S. M. Homayouni, “Joint production and transportation scheduling in flexible manufacturing systems,” *J. Glob. Optim.*, pp. 1–30, 2018.

[164] P. Lacomme, M. Larabi, and N. Tchernev, “Job-shop based framework for simultaneous scheduling of machines and automated guided vehicles,” *Int. J. Prod. Econ.*, vol. 143, no. 1, pp. 24–34, 2013.

[165] M. H. F. Bin Md Fauadi and T. Murata, “Makespan minimization of machines and Automated Guided Vehicles schedule using Binary Particle Swarm Optimization BT - International MultiConference of Engineers and Computer Scientists 2010, IMECS 2010, March 17, 2010 - March 19, 2010,” vol. III, pp. 1897–1902, 2010.

[166] L. Deroussi and S. Norre, “Simultaneous scheduling of machines and vehicles for the flexible job shop problem Solution approach : basic ideas,” *Int. Conf. Metaheuristics Nat. Inspired Comput.*, no. February, pp. 2–3, 2010.

[167] M. V. S. Kumar, R. Janardhana, and C. S. P. Rao, “Simultaneous scheduling of machines and vehicles in an FMS environment with alternative routing,” *Int. J. Adv. Manuf. Technol.*, vol. 53, no. 1–4, pp. 339–351, 2011.

[168] C. Sahin, M. Demirtas, R. Erol, A. Baykasoğlu, and V. Kaplanoğlu, “A multi-agent based approach to dynamic scheduling with flexible processing capabilities,” *J. Intell. Manuf.*, vol. 28, no. 8, pp. 1827–1845, 2017.

[169] N. Howden, R. Rönnquist, A. Hodgson, and A. Lucas, “JACK Intelligent Agents - Summary of an Agent Infrastructure,” in *Proceedings of the 5th International Conference on Autonomous Agents*, 2001.

[170] J. T. Lin, C. C. Chiu, and Y. H. Chang, “Simulation-based optimization approach for simultaneous scheduling of vehicles and machines with processing time uncertainty in FMS,” *Flex. Serv. Manuf. J.*, pp. 1–38, 2017.

[171] Q. Zhang, H. Manier, and M.-A. Manier, “Metaheuristics for Job Shop Scheduling with Transportation,” in *Metaheuristics for Production Scheduling*, 2013, pp. 465–493.

[172] L. Deroussi, “A Hybrid PSO Applied to the Flexible Job Shop with Transport,” in *Swarm Intelligence Based Optimization*, 2014, pp. 115–122.

[173] H. E. Nouri, O. Belkahla Driss, and K. Ghédira, “Simultaneous scheduling of machines and transport robots in flexible job shop environment using hybrid metaheuristics based on clustered holonic multiagent model,” *Comput. Ind. Eng.*, vol. 102, pp. 488–501, Dec. 2016.

[174] S. M. Homayouni and D. B. M. M. Fontes, “Joint scheduling of production and transport with alternative job routing in flexible manufacturing systems,” *AIP Conf. Proc.*, vol. 2070, no. January, 2019.

[175] A. Ham, “Transfer-robot task scheduling in job shop,” *Int. J. Prod. Res.*, vol. 0, no. 0, pp. 1–11, 2020.

[176] A. Ham, “Transfer-robot task scheduling in flexible job shop,” *J. Intell. Manuf.*, no. February, 2020.

[177] M. L. Junior and M. G. Filho, “Production planning and control for remanufacturing: literature review and analysis,” *Prod. Plan. Control*, vol. 23, no. 6, pp. 419–435, 2012.

[178] J. Jungbluth, “ENTWICKLUNG EINES INTELLIGENTEN, ROBOTERGESTÜTZTEN ASSISTENZSYSTEMS FÜR DIE DEMONTAGE INDUSTRIELLER PRODUKTE,” University of Luxembourg, 2019.

[179] P. Laborie, “Introduction to CP Optimizer for Scheduling,” *IBM*, 2016. [Online]. Available: https://www.researchgate.net/publication/317932913_A_Not_So_Short_Introduction_to_CP_Optimizer_for_Scheduling. [Accessed: 12-Jun-2019].

[180] 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.

[181] “MiR100™ | Mobile Industrial Robots.” [Online]. Available: https://www.mobile-industrial-robots.com/de/products/mir100/. [Accessed: 04-Feb-2020].

[182] P. Laborie and D. Godard, “Self-adapting large neighborhood search: Application to single-mode scheduling problems,” *Proc. MISTA-07, Paris*, p. 8, 2007.

[183] R. Erol, C. Sahin, A. Baykasoglu, and V. Kaplanoglu, “A multi-agent based approach to dynamic scheduling of machines and automated guided vehicles in manufacturing systems,” *Appl. Soft Comput.*, vol. 12, no. 6, pp. 1720–1732, 2012.

[184] Y. Zheng, Y. Xiao, and Y. Seo, “A tabu search algorithm for simultaneous machine/AGV scheduling problem,” *Int. J. Prod. Res.*, vol. 52, no. 19, pp. 5748–5763, 2014.

[185] S. Huang, “Optimization of Job Shop Scheduling with Material Handling by Automated Guided Vehicle,” *ProQuest Diss. Theses*, p. 92, 2018.

[186] U. Brecht, *BWL für Führungskräfte: Was Entscheider im Unternehmen wissen müssen*. Springer Fachmedien Wiesbaden, 2012.

[187] E. Geisberger, M. Cengarle, P. Keil, J. Niehaus, C. Thiel, and H.-J. Thönnißen-Fires, “Innovationsmotor für Mobilität, Gesundheit, Energie und Produktion,” *Acatech*, p. 35, 2011.

[188] C. Böning, “MES | Trends, Lösungen und Umsetzung von ME-Systemen,” *Institut für integrierte Produktion Hannover*. [Online]. Available: https://www.iph-hannover.de/de/information/erp-mes/mes/#implement. [Accessed: 20-Oct-2020].

[189] S. Groß, W. Gerke, and P. Plapper, “Human-Robot-Collaboration for dismantling processes,” in *Robotix-Academy Conference for Industrial Robotics (RACIR)*, 2017, pp. 9–12.

[190] S. Groß, W. Gerke, and P. Plapper, “Mensch-Roboter-Kollaboration für Demontageprozesse,” in *AALE 2018*, 2018th ed., Köln: VDE-Verlag, 2018, pp. 327–337.

[191] S. Groß, W. Gerke, and P. Plapper, “A survey: Scheduling of Automated Guided Vehicles in Flexible (Re-)Manufacturing Systems,” in *Robotix-Academy Conference for Industrial Robotics (RACIR) 2019*, 2019.

[192] S. Groß, W. Gerke, and P. Plapper, “Simulation-based optimization using multi-agent technology for efficient and flexible production planning and control in remanufacturing,” in *International Conference on Remanufacturing 2019*, 2019.

[193] S. Groß, W. Gerke, and P. Plapper, “Optimized and flexible scheduling of AGVs and process machines in Remanufacturing 4.0 Systems using multi-agent technology and simultaneous scheduling,” in *Abstracts III International Workshop on Autonomous Remanufacturing*, F. J. Ramírez Fernández and A. Honrubia Escribano, Eds. 2019, p. 23.

[194] S. Groß, W. Gerke, and P. Peter, “Agentenbasierte, hybride Steuerungsarchitektur für cyberphysische Refabrikationssysteme,” in *Tagungsband AALE 2020*, 2020.

[195] S. Groß, T. Bartscherer, A. Pereira, W. Gerke, and P. Plapper, “Mensch-Roboter-Kollaboration

in der Domäne Refabrikation – State-of-the-Art und Ausblick," in *Tagungsband AALE 2020*, 2020.

[196] S. Groß, W. Gerke, and P. Plapper, "Agent-based, hybrid control architecture for optimized and flexible production scheduling and control in remanufacturing," *J. Remanufacturing*, 2020.

[197] M. Schulz and B. Della Vedova, "Richtlinie 2014/95/EU des Europäischen Parlaments und des Rates vom 22. Oktober 2014 zur Änderung der Richtlinie 2013/34/EU im Hinblick auf die Angabe nichtfinanzieller und die Diversität betreffender Informationen durch bestimmte große Unternehmen und Gruppen Text von Bedeutung für den EWR," *Europäischen Union*, 2014. [Online]. Available: https://eur-lex.europa.eu/legal-content/DE/TXT/?uri=CELEX:32014L0095. [Accessed: 06-May-2019].

[198] B. Burger, "Öffentliche Nettostromerzeugung in Deutschland im Jahr 2018," Freiburg, 2019.

[199] D. Lamm, "Leistungselektronik," *EVA Fahrzeugtechnik GmbH*. [Online]. Available: https://www.evafahrzeugtechnik.de/kernkompetenzen/elektromobilitaet/leistungselektronik.html. [Accessed: 09-May-2019].

[200] "Kraftfahrt-Bundesamt - Neuzulassungen - Jahresbilanz der Neuzulassungen 2018," *Kraftfahrt-Bundesamt*, 2019. [Online]. Available: https://www.kba.de/DE/Statistik/Fahrzeuge/Neuzulassungen/n_jahresbilanz.html?nn=644522. [Accessed: 09-May-2019].

[201] "Elektromobilität in Deutschland - VDA," *Verband der Automobilindustrie e.V.*, 2019. [Online]. Available: https://www.vda.de/de/themen/innovation-und-technik/elektromobilitaet/elektromobilitaet-in-deutschland.html. [Accessed: 09-May-2019].

[202] P. Bartel, "Aktuelle Herausforderungen und Strategien des Remanufacturing von Automobilteilen," *Robert Bosch GmbH*, 2016. [Online]. Available: http://docplayer.org/6623127-Aktuelle-herausforderungen-und-strategien-des-remanufacturing-von-automobilteilen-peter-bartel-robert-bosch-gmbh.html. [Accessed: 07-May-2019].

[203] D. Parker *et al.*, "Remanufacturing Market Study," *Eur. Remanufacturing Netw.*, no. 645984, p. 145, 2015.

[204] W. M. Hauser and R. T. Lund, *The Remanufacturing Industry: Anatomy of a Giant ; a View of Remanufacturing in America Based on a Comprehensive Survey Across the Industry*. Department of Manufacturing Engineering, Boston Univ., 2003.

[205] R. Steinhilper and S. Freiberger, "Neue Qualifikation zur Kfz-Ersatzteilversorgung durch Austausch- teileproduktion (Remanufacturing) von mechatronischen Baugrup-," *Berufsbildungswissenschaftliche Schriften*, pp. 89–98, 2010.

[206] N. Nasr and M. Thurston, "Remanufacturing: A Key Enabler to Sustainable Product Systems," *Proc. LCE 2006*, no. January 2006, pp. 15–18, 2006.

[207] U. Lange, *Ressourceneffizienz durch Remanufacturing - Industrielle Aufarbeitung von Altteilen*, no. 18. VDI Zentrum Ressourceneffizienz GmbH, 2017.

[208] B. Korte, J. Vygen, U. Brenner, and R. Randow, *Kombinatorische Optimierung: Theorie und Algorithmen*. Springer Berlin Heidelberg, 2018.

[209] S. Noor, M. I. Lali, and M. S. Nawaz, "SOLVING JOB SHOP SCHEDULING PROBLEM WITH GENETIC ALGORITHM," *Sci. Int.*, vol. 27, pp. 3367–3371, 2015.

[210] K.-M. Lee and T. Yamakawa, “A genetic algorithm for general machine scheduling problems,” *1998 Second Int. Conf. Knowledge-Based Intell. Electron. Syst. Proc. KES’98 (Cat. No. 98EX111)*, vol. 2, pp. 60–66, 1998.

[211] D. Pavlyuk, “Feature selection and extraction in spatiotemporal traffic forecasting: a systematic literature review,” *Eur. Transp. Res. Rev.*, vol. 11, 2019.

[212] P. Vilím, P. Laborie, and P. Shaw, “Failure-Directed Search for Constraint-Based Scheduling,” *Int. Conf. AI OR Tech. Constraint Program. Comb. Optim. Probl.*, pp. 437–453, 2015.

[213] D. B. M. M. Fontes and S. M. Homayouni, “Joint production and transportation scheduling in flexible manufacturing systems,” *J. Glob. Optim.*, no. June 2018, pp. 1–30, 2018.

[214] B. Somanath, K. Mallikarjuna, Y. Samson, and K. R. Srinivas, “Differential Evolution and Simulated Annealing Technique for Design and Developing of Process Plan to Perform Optimization on Finite Capacity Scheduling for Simultaneous System of Machines,” *Int. Res. J. Eng. Technol.*, vol. 4, no. 3, pp. 738–742, 2017.

[215] P. B. Kanakavalli, V. B. Vommi, and N. R. Medikondu, “Fuzzy heuristic algorithm for simultaneous scheduling problems in flexible manufacturing system,” *Manag. Sci. Lett.*, vol. 8, no. 12, pp. 1319–1330, 2018.

[216] V. K. Chawla, A. K. Chanda, and S. Angra, “The scheduling of automatic guided vehicles for the workload balancing and travel time minimi-zation in the flexible manufacturing system by the nature-inspired algorithm,” *J. Proj. Manag.*, vol. 4, pp. 19–30, 2018.

[217] V. K. Chawla, A. K. Chanda, and S. Angra, “Simultaneous workload balancing and travel time minimization of automatic guided vehicles,” *J. Phys. Conf. Ser.*, vol. 1240, no. 1, 2019.

[218] P. R and S. A, “Utilization of AGVs and Machines in FMS Environment,” *J. Mater. Sci. Eng.*, vol. 5, no. 4, 2016.

[219] D. Tang, K. Zheng, and W. Gu, “Hormone Regulation Based Approach for Distributed and On-line Scheduling of Machines and AGVs,” in *Adaptive Control of Bio-Inspired Manufacturing Systems*, Singapore: Springer Singapore, 2020, pp. 47–72.

[220] L. Deroussi, “A Hybrid PSO Applied to the Flexible Job Shop with Transport,” in *International Conference on Swarm Intelligence Based Optimization*, 2014, no. Mai, pp. 115–122.

[221] R. Smith, “The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver,” *IEEE Trans. Comput.*, vol. 29, no. 12, pp. 1104–1113, 1980.


Interreg
EUROPEAN UNION
Grande Région | Großregion
Robotix-Academy
Fonds européen de développement régional | Europäischer Fonds für regionale Entwicklung


uni.lu
UNIVERSITÉ DU
LUXEMBOURG


TRIER UNIVERSITY
OF APPLIED SCIENCES
Umwelt-Campus Birkenfeld
Environmental Campus Birkenfeld